OS/VS2 Development Series
Sample S/370 Assembler Programs

August 8, 2010

David J. Walling
david.walling@yahoo.com

Contents

Introduction

About This Series

This series of documents describes the installation, configuration and use of an OS/VS2 Release 3.8 computing environment for software development, suitable for adapting to later generations of this operating system while retaining backward compatibility with older systems.

About This Document

This document provides several beginning sample programs that illustrate core concepts in structuring S/380 Assembly language programs for OS/VS2.

Intended Readership

This document is intended for those interested in developing software for the mainframe using contemporary operating systems descended from OS/VS2 or legacy operation systems such as version 3.8. This document may also be of use to those interested in learning more about how OS/VS2 is installed, configured and maintained.

About the Author

David J. Walling has developed commercial software professionally for various operating platforms including mainframes, minis and microcomputers since 1986 and may be contacted by email at david.walling@yahoo.com.

Sample Programs - Part One

The following three sample programs introduce S/370 Assembler programming styles.

ASM001

ASM001 - Source Code

```
000001 ************************************************************************
000002 *
000003 *        PROGRAM    ASM001
000004 *
000005 *        SAMPLE ASSEMBLY-LANGUAGE PROGRAM ILLUSTRATING THE USE OF
000006 *        THE USING, CSECT AND END STATEMENTS. THIS PROGRAM ISSUES
000007 *        A 'WTO' MACRO TO SEND A MESSAGE TO THE OPERATOR CONSOLE
000008 *        AND RETURNS TO THE CALLING PROGRAM (MVS).
000009 *
000010 *        ENTRY       1     PROGRAM ARGUMENTS ADDRESS
000011 *                    13    CALLER SAVE AREA ADDRESS
000012 *                    14    RETURN ADDRESS
000013 *                    15    CONTROL SECTION ADDRESS
000014 *
000015 *        EXIT        14    RETURN ADDRESS
000016 *
000017 ************************************************************************
000018          USING $CODE,15                RESOLVE CODE TO REG 15
000019 $CODE     CSECT                         PROGRAM ENTRY
000020          WTO   'HELLO, WORLD!'         DISPLAY MESSAGE TO SYSOP
000021          BR    14                      RETURN TO CALLER
000022          END   $CODE
```

This sample S/370 program issues a WTO macro to display a message on the operator console. Our source-code samples will begin with a "flower-box" of comments that includes a description of the program's function and the contents of general purpose registers on entry and exit.

ASM001 - Job-Control Language

The JCL to assemble, link and execute the sample program ASM001 is shown below. The job is named JASM001. The three job steps are named JS010, JS020 and JS030. Note that the source, object and executable names are all ASM001. We will see, later, how to define this JCL as a catalogued procedure and pass the name of the source member as a parameter in order to reuse this JCL for assembling any program.

Note also that the EXEC statements for JS020 and JS030 are both given a COND parameter with a value of '(0,LT)'. This will instruct the Job Entry System (JES) to run JS020 unless JS010 completes with a result code greater than 0. Similarly, JES will run JS030 unless either JS010 or JS020 complete with a result code greater than 0.

```
000001 //JASM001  JOB (001),'DWALL01',CLASS=A,MSGCLASS=A,
000002 //             MSGLEVEL=(1,1),NOTIFY=DWALL01
000003 //*
000004 //*            ASSEMBLE THE PROGRAM
000005 //*
000006 //JS010   EXEC PGM=IFOX00,REGION=256K,
000007 //             PARM='OBJ,RENT,XREF'
000008 //SYSLIB    DD DISP=SHR,DSN=SYS1.MACLIB
000009 //           DD DISP=SHR,DSN=DWALL01.SEVYN.MACLIB
000010 //SYSUT1    DD DSN=&&SYSUT1,UNIT=SYSSQ,
000011 //             SPACE=(1700,(600,100)),SEP=(SYSLIB)
000012 //SYSUT2    DD DSN=&&SYSUT2,UNIT=SYSSQ,
000013 //             SPACE=(1700,(300,50)),SEP=(SYSUT1)
000014 //SYSUT3    DD DSN=&&SYSUT3,UNIT=SYSSQ,
000015 //             SPACE=(1700,(300,40))
000016 //SYSPRINT  DD SYSOUT=A,DCB=BLKSIZE=1089
000017 //SYSPUNCH  DD SYSOUT=B
000018 //SYSGO     DD DISP=SHR,DSN=DWALL01.SEVYN.OBJLIB(ASM001)
000019 //SYSIN     DD DISP=SHR,DSN=DWALL01.SEVYN.SORLIB(ASM001)
000020 //*
000021 //*            LINK THE PROGRAM
000022 //*
000023 //JS020   EXEC PGM=IEWL,REGION=256K,COND=(0,LT),
000024 //             PARM='XREF,LET,LIST,NCAL,RENT'
000025 //SYSUT1    DD DSN=&&SYSUT1,UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),
000026 //             SPACE=(1024,(50,20))
000027 //SYSPRINT  DD SYSOUT=A
000028 //SYSLMOD   DD DISP=SHR,DSN=DWALL01.SEVYN.MODLIB(ASM001)
000029 //SYSLIN    DD DISP=SHR,DSN=DWALL01.SEVYN.OBJLIB(ASM001)
000030 //SYSIN     DD *
000031 NAME ASM001(R)
000032 /*
000033 //*
000034 //*            RUN THE PROGRAM
000035 //*
000036 //JS030   EXEC PGM=ASM001,REGION=256K,COND=(0,LT)
000037 //STEPLIB   DD DISP=SHR,DSN=DWALL01.SEVYN.MODLIB
000038 //SYSPRINT  DD SYSOUT=*
000039 //SYSUDUMP  DD SYSOUT=*
000040 //SYSABEND  DD SYSOUT=*
000041 //
```

ASM001 - System Log

```
16.14.35 JOB   37  $HASP100 JASM001  ON INTRDR      DWALL01
16.14.35 JOB   37  $HASP373 JASM001  STARTED - INIT  1 - CLASS A - SYS BSP1
16.14.35 JOB   37  IEF403I JASM001 - STARTED - TIME=16.14.35
16.14.35 JOB   37  IEFACTRT - Stepname  Procstep  Program   Retcode
16.14.35 JOB   37  JASM001    JS010               IFOX00    RC= 0000
16.14.35 JOB   37  JASM001    JS020               IEWL      RC= 0000
16.14.35 JOB   37  +HELLO, WORLD!
16.14.35 JOB   37  JASM001    JS030               ASM001    RC= 0000
16.14.35 JOB   37  IEF404I JASM001 - ENDED - TIME=16.14.35
16.14.35 JOB   37  $HASP395 JASM001  ENDED
16.14.35           $HASP309    INIT  1 INACTIVE ******** C=A
16.14.35 JOB   37  $HASP150 JASM001  ON PRINTER1       214 LINES
16.14.35           $HASP160 PRINTER1 INACTIVE - CLASS=A
```

The system log for JASM001 is shown above. JES assigns the job number 37 and places the job on the initiator reader. Initiator 1 accepts the job in class A and starts the job. JS010 and JS020 complete with return code 0000. JS030 then executes and issues the SVC 35 call to send the message, "HELLO, WORLD!" to the operator console. JS030 completes with return code 0000.

JES reaches the end of the JCL stream. Initiator 1 becomes inactive until the next job is available. 214 output lines from the job stream are directed to the PRINTER1 device.

ASM001 Assembler Listing

```
                                      1 ********************************************************************
                                      2 *
                                      3 *        PROGRAM    ASM001
                                      4 *
                                      5 *        SAMPLE ASSEMBLY-LANGUAGE PROGRAM ILLUSTRATING THE USE OF
                                      6 *        THE USING, CSECT AND END STATEMENTS. THIS PROGRAM ISSUES
                                      7 *        A 'WTO' MACRO TO SEND A MESSAGE TO THE OPERATOR CONSOLE
                                      8 *        AND RETURNS TO THE CALLING PROGRAM (MVS).
                                      9 *
                                     10 *        ENTRY       1     PROGRAM ARGUMENTS ADDRESS
                                     11 *                    13     CALLER SAVE AREA ADDRESS
                                     12 *                    14     RETURN ADDRESS
                                     13 *                    15     CONTROL SECTION ADDRESS
                                     14 *
                                     15 *        EXIT        14     RETURN ADDRESS
                                     16 *
                                     17 ********************************************************************
                           00000     18          USING $CODE,15                RESOLVE CODE TO REG 15
000000                               19 $CODE    CSECT                         PROGRAM ENTRY
                                     20          WTO   'HELLO, WORLD!'         DISPLAY MESSAGE TO SYSOP
000000                               21+         CNOP  0,4                                                  09800002
000000 4510 F016      00016          22+         BAL   1,IHB0001A                      BRANCH AROUND MESSAGE  09850002
000004 0011                          23+         DC    AL2(17)             TEXT LENGTH                      13200002
000006 0000                          24+         DC    B'0000000000000000' MCS FLAGS                        13250002
000008 C8C5D3D3D66B40E6              25+         DC    C'HELLO, WORLD!'                                     13350002
000016                               26+IHB0001A DS    0H                                                   15850002
000016 0A23                          27+         SVC   35                                                   15950002
000018 07FE                          28          BR    14                      RETURN TO CALLER
000000                               29          END   $CODE
```

ASM001 - Notes

Upon entry to a program, general register 15 holds the address of the program control section. Therefore, we want the assembler to resolve address constants to base-displacement encoding using register 15. The 'USING $CODE,15' statement on line 18 directs the assembler to generate a base-displacement encoding of 'F016' for the BAL instruction on line 22 which references the symbol IHB0001A.

Also upon entry to the program, general register 14 contains the address of the next sequential instruction in the calling program. In other words, the address in register 14 is where we want to direct control at the end of our sample program. This is accomplished by the branch (BR 14) instruction on line 28.

The WTO macro is expanded into a Branch-And-Link (BAL) instruction which alters the contents of register 1, several Define-Constant (DC) instructions and a Supervisor Call (SVC) instruction. Since the SVC call does not alter register 14, it is not necessary here to save register 14 prior to issuing the WTO macro. However, some macros do alter general purposes registers in use by our sample programs. Therefore our next samples will introduce standard methods for saving general purpose registers between calls to operating system services and our own subroutines. These samples will also introduce standard usages for general purpose registers.

ASM002

ASM002 - Source Code

```
000001 ************************************************************************
000002 *
000003 *        PROGRAM    ASM002
000004 *
000005 *        SAMPLE ASSEMBLY-LANGUAGE PROGRAM ILLUSTRATING THE USE OF
000006 *        THE COPY, SPACE, EJECT AND LTORG STATEMENTS. THIS PROGRAM
000007 *        BUILDS ON THE SAMPLE PROGRAM ASM001, ADDING LOGIC TO SAVE
000008 *        CALLING PROGRAM REGISTERS ON ENTRY, RESTORING REGISTER
000009 *        VALUES ON EXIT AND SETTING A RETURN CODE.
000010 *
000011 *        ENTRY       1     PROGRAM ARGUMENTS ADDRESS
000012 *                   13     CALLER SAVE AREA ADDRESS
000013 *                   14     RETURN ADDRESS
000014 *                   15     CONTROL SECTION ADDRESS
000015 *
000016 *        EXIT       15     RETURN CODE
000017 *
000018 ************************************************************************
000019          EJECT
000020          COPY  MEQUREGS
000021          SPACE
000022          USING $CODE,R11               RESOLVE CODE TO REG 11
000023          EJECT
000024 $CODE    CSECT                         PROGRAM ENTRY
000025          SPACE
000026          STM   R14,R12,12(R13)         SAVE CALLER REG 14-15,0-12
000027          LR    R11,R15                 LOAD 1ST CSECT BASE ADDR
000028          SPACE
000029          WTO   'HELLO, WORLD!'         DISPLAY MESSAGE TO SYSOP
000030          SPACE
000031          XR    R15,R15                 ZERO RETURN CODE
000032          SPACE
000033          LM    R0,R12,20(R13)          RESTORE CALLER REG 0-12
000034          L     R14,12(R13)             RESTORE CALLER REG 14
000035          BR    R14                     RETURN TO CALLER
000036          EJECT
000037          LTORG
000038          EJECT
000039          END   $CODE
```

Sample program ASM002 introduces the use of the COPY, EJECT, SPACE and LTORG statements. This sample also extends our source-code to save the general purpose values passed by the calling program and restoring these register values on return.

ASM002 - Job Control

The Job Control stream for ASM002 is the same as that for ASM001 except that the program name ASM001 has been replaced with ASM002.

ASM002 - System Log

The System Log output for ASM002 is the same as that for ASM002

ASM002 - Assembler Listing

```
                                 1 ************************************************************************
                                 2 *
                                 3 *        PROGRAM    ASM002
                                 4 *
                                 5 *        SAMPLE ASSEMBLY-LANGUAGE PROGRAM ILLUSTRATING THE USE OF
                                 6 *        THE COPY, SPACE, EJECT AND LTORG STATEMENTS. THIS PROGRAM
                                 7 *        BUILDS ON THE SAMPLE PROGRAM ASM001, ADDING LOGIC TO SAVE
                                 8 *        CALLING PROGRAM REGISTERS ON ENTRY, RESTORING REGISTER
                                 9 *        VALUES ON EXIT AND SETTING A RETURN CODE.
                                10 *
                                11 *        ENTRY       1     PROGRAM ARGUMENTS ADDRESS
                                12 *                    13     CALLER SAVE AREA ADDRESS
                                13 *                    14     RETURN ADDRESS
                                14 *                    15     CONTROL SECTION ADDRESS
                                15 *
                                16 *        EXIT        15     RETURN CODE
                                17 *
                                18 ************************************************************************
                                                                                                  PAGE    3

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                 ASM 0201 13.40 05/15/10

                                   20          COPY  MEQUREGS

                          00000    22 R0       EQU   0                      SYSTEM/TEMP USE
                          00001    23 R1       EQU   1                      SYSTEM/TEMP USE
                          00002    24 R2       EQU   2                      VOLATILE/PARAMETER
                          00003    25 R3       EQU   3                      VOLATILE/PARAMETER
                          00004    26 R4       EQU   4                      NON-VOLATILE
                          00005    27 R5       EQU   5                      NON-VOLATILE
                          00006    28 R6       EQU   6                      NON-VOLATILE
                          00007    29 R7       EQU   7                      NON-VOLATILE
                          00008    30 R8       EQU   8                      NON-VOLATILE
                          00009    31 R9       EQU   9                      NON-VOLATILE
                          0000A    32 R10      EQU   10                     CSECT BASE ADDR 2
                          0000B    33 R11      EQU   11                     CSECT BASE ADDR 1
                          0000C    34 R12      EQU   12                     DSECT BASE ADDR
                          0000D    35 R13      EQU   13                     SAVE AREA ADDR
                          0000E    36 R14      EQU   14                     LINKAGE/RETURN ADDR
                          0000F    37 R15      EQU   15                     LINKAGE/ENTRY ADDR/RESULT CODE
```

```
                                                                                                                  PAGE    4
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                          ASM 0201 13.40 05/15/10

                           00000    40          USING $CODE,R11               RESOLVE CODE TO REG 11
                                                                                                                  PAGE    5
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                          ASM 0201 13.40 05/15/10

000000                               42 $CODE    CSECT                         PROGRAM ENTRY

000000 90EC D00C      0000C          44          STM   R14,R12,12(R13)         SAVE CALLER REG 14-15,0-12
000004 18BF                          45          LR    R11,R15                 LOAD 1ST CSECT BASE ADDR

                                     47          WTO   'HELLO, WORLD!'         DISPLAY MESSAGE TO SYSOP
000006 0700                          48+         CNOP  0,4                                                    09800002
000008 4510 B01E      0001E          49+         BAL   1,IHB0001A                        BRANCH AROUND MESSAGE  09850002
00000C 0011                          50+         DC    AL2(17)             TEXT LENGTH                      13200002
00000E 0000                          51+         DC    B'0000000000000000' MCS FLAGS                        13250002
000010 C8C5D3D3D66B40E6              52+         DC    C'HELLO, WORLD!'                                     13350002
00001E                               53+IHB0001A DS    0H                                                   15850002
00001E 0A23                          54+         SVC   35                                                   15950002

000020 17FF                          56          XR    R15,R15                 ZERO RETURN CODE

000022 980C D014      00014          58          LM    R0,R12,20(R13)          RESTORE CALLER REG 0-12
000026 58ED 000C      0000C          59          L     R14,12(R13)             RESTORE CALLER REG 14
00002A 07FE                          60          BR    R14                     RETURN TO CALLER
                                                                                                                  PAGE    6
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                          ASM 0201 13.40 05/15/10

000030                               62          LTORG
                                                                                                                  PAGE    7
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                          ASM 0201 13.40 05/15/10

000000                               64          END   $CODE
```

SYMBOL LEN VALUE DEFN REFERENCES ASM 0201 13.40 05/15/10

```
$CODE    00001 00000000 00042  00040 00064
IHB0001A 00002 0000001E 00053  00049
R0       00001 00000000 00022  00058
R1       00001 00000001 00023
R10      00001 0000000A 00032
R11      00001 0000000B 00033  00040 00045
R12      00001 0000000C 00034  00044 00058
R13      00001 0000000D 00035  00044 00058 00059
R14      00001 0000000E 00036  00044 00059 00060
R15      00001 0000000F 00037  00045 00056 00056
R2       00001 00000002 00024
R3       00001 00000003 00025
R4       00001 00000004 00026
R5       00001 00000005 00027
R6       00001 00000006 00028
R7       00001 00000007 00029
R8       00001 00000008 00030
R9       00001 00000009 00031
```

ASM002 - Notes

In ASM002, we begin to refer to general purpose registers using symbolic names instead of numbers. So register 14 is labeled "R14" instead of just "14". This has no effect on the machine code generated by the assembler. However it does improve readability by making it clear when a register is being used instead of a decimal literal. So, for example, on line 33 above, "20" is clearly a decimal displacement from the base register "R13". Using symbolic names for registers also allows us to track the usage of general purpose registers in the symbol cross-references generated by the assembler.

On line 20 above, we use the COPY statement to include an external source-code file in the assembly of our source-code. The operand "MEQUREGS" must be the name of a PDS member found in the SYSLIB concatenation. The annotated source-code listing produced by the assembler that shows generated machine code also shows the contents of these "copy book" members.

The SPACE and EJECT statements can improve the readability of source-code listings produced by the assembler by inserting blank lines or moving to the start of the next logical page of output.

The LTORG statement directs the assembler to emit Define Constant (DC) statements in which to hold literal values defined in the source code. So, for example, an instruction, "MVI WXCODE,=X'FF'", would require the literal constant, X'FF', to be defined somewhere in the control section. When the assembler encounters a LTORG statement in a control section, it generates constant definitions for literals. In this sample program, ASM002, no literal constants are used, so the LTORG statement does not generate any Define Constant statements.

At the beginning of the $CODE control section, ASM002 saves the values of the general purpose registers passed to it from the calling program. To do this, ASM2 uses the save area provided by the calling program at the address in general register 13. By convention, registers are stored beginning with register 14 and 15, followed by registers 0 through 12. These registers are stored at displacement 12 off of register 13. The first 12 bytes, or 3 fullwords, are reserved for holding other address information as we will see below. After saving the calling program's general purpose registers, ASM002 loads register 11 with the address of the control section in register 15. By convention, our programs will use register 11 as the default control section base register. In order to do this, a "USING $CODE,R11" statement is required prior to any statement that will require the assembler to generate a base-displacement address for a symbol in the control section.

After the WTO macro, ASM002 resets register 15 to zero and will use this register to return the program result code. Then, just prior to returning to the calling program, our calling program's registers are restored by loading them back out of the storage area at the address in register 13. Note that register 15 is not loaded since it already holds the result code we want ASM002 to return to the caller.

ASM003

ASM003 - Source Code

```
000001 ************************************************************************
000002 *
000003 *        PROGRAM    ASM003
000004 *
000005 *        SAMPLE ASSEMBLY-LANGUAGE PROGRAM ILLUSTRATING THE USE OF
000006 *        THE GETMAIN AND FREEMAIN MACROS. THIS PROGRAM BUILDS ON
000007 *        THE SAMPLE PROGRAM ASM002, ADDING LOGIC TO ALLOCATE LOCAL
000008 *        STORAGE AND CALL SUBROUTINES.
000009 *
000010 *        ENTRY       1     PROGRAM ARGUMENTS ADDRESS
000011 *                   13     CALLER SAVE AREA ADDRESS
000012 *                   14     RETURN ADDRESS
000013 *                   15     CONTROL SECTION ADDRESS
000014 *
000015 *        EXIT       15     RETURN CODE
000016 *
000017 ************************************************************************
000018          EJECT
000019          COPY  MEQUREGS
000020          SPACE
000021 ************************************************************************
000022 *
000023 *        USINGS
000024 *
000025 ************************************************************************
000026          SPACE
000027          USING $CODE,R11,R10            RESOLVE CODE TO REG 11,10
000028          USING $DATA,R13                RESOLVE DATA TO REG 13
000029          EJECT
000030 ************************************************************************
000031 *
000032 *        $CODE                          PROGRAM CODE SECTION
000033 *
000034 ************************************************************************
000035          SPACE
000036 $CODE    CSECT                          PROGRAM ENTRY
000037          SPACE
000038 *                                       ******************************
000039 *                                       *   SAVE CALLER REGS
000040 *                                       *   SETUP BASE REGS
000041 *                                       *   ALLOCATE STORAGE
000042 *                                       ******************************
000043          SPACE
000044          STM   R14,R12,12(R13)          SAVE CALLER REG 14-15,0-12
000045          LR    R11,R15                  LOAD 1ST CSECT BASE ADDR
000046          LA    R15,1                    ONE
000047          LA    R10,4095(R15,R11)        LOAD 2ND CSECT BASE ADDR
000048          SPACE
000049          LA    R0,E$DATAL               LENGTH OF STORAGE
000050          SPACE
000051          GETMAIN R,LV=(0)               ALLOCATE STORAGE
000052          SPACE
000053          LA    R15,ERNOMEM              ASSUME GETMAIN ERROR
000054          LTR   R1,R1                    IF MEMORY ALLOCATED
000055          BZ    @CODE20
000056          SPACE
```

```
000057 *                                      ****************************
000058 *                                      *  RESET STORAGE TO NULLS
000059 *                                      *  SETUP STORAGE BASE ADDR
000060 *                                      ****************************
000061          SPACE
000062          LR    R2,R1                   TARGET ADDR IS STORAGE
000063          LA    R3,E$DATAL              LENGTH IS STORAGE LENGTH
000064          XR    R4,R4                   SOURCE ADDR IS NULL
000065          LR    R5,R4                   SOURCE LENGTH IS ZERO (PAD)
000066          MVCL  R2,R4                   ZERO STORAGE
000067          SPACE
000068          ST    R13,4(R1)               SAVE CALLER STORAGE ADDR
000069          ST    R1,8(R1)                SAVE OUR STORAGE ADDR
000070          LR    R13,R1                  LOAD STORAGE BASE REG
000071          SPACE
000072 *                                      ****************************
000073 *                                      *  DO PROGRAM INITIALIZATION
000074 *                                      *  DO PROGRAM MAINLINE
000075 *                                      *  DO PROGRAM FINALIZATION
000076 *                                      ****************************
000077          SPACE
000078          XR    R15,R15                 ZERO RETURN CODE
000079          SPACE
000080          BAL   R14,@INIT               DO INITIALIZATION
000081          SPACE
000082          LTR   R15,R15                 IF INITIALIZATION OK
000083          BNZ   @CODE10
000084          SPACE
000085          BAL   R14,@MAIN                 DO MAINLINE
000086          SPACE
000087 @CODE10  DS    0H                      END IF
000088          SPACE
000089          BAL   R14,@FINAL              DO FINALIZATION
000090          SPACE
000091 *                                      ****************************
000092 *                                      *  RESTORE CALLER SAVE AREA
000093 *                                      *  RELEASE STORAGE
000094 *                                      ****************************
000095          SPACE
000096          LR    R1,R13                  OUR SAVE AREA ADDR
000097          L     R13,4(R1)               CALLER SAVE AREA ADDR
000098          SPACE
000099          LA    R0,E$DATAL              LENGTH OF SAVE AREA
000100          SPACE
000101          FREEMAIN R,LV=(0),A=(R1)      RELEASE STORAGE
000102          SPACE
000103 @CODE20  DS    0H                    END IF
000104          SPACE
000105 *                                    ****************************
000106 *                                    *  RESTORE CALLER REGS
000107 *                                    *  RETURN TO CALLER
000108 *                                    ****************************
000109          SPACE
000110          LM    R0,R12,20(R13)        RESTORE CALLER REG 0-12
000111          L     R14,12(R13)           RESTORE CALLER REG 14
000112          BR    R14                   RETURN TO CALLER
000113          EJECT
```

```
000114 ********************************************************************
000115 *
000116 *        @INIT                          INTIIALIZE PROGRAM
000117 *
000118 ********************************************************************
000119          SPACE
000120 @INIT    DS    0H
000121          SPACE
000122          ST    R14,W@INIT               SAVE RETURN ADDR
000123          SPACE
000124          SPACE
000125          L     R14,W@INIT               LOAD RETURN ADDR
000126          BR    R14                      RETURN TO CALLER
000127          EJECT
000128 ********************************************************************
000129 *
000130 *        @MAIN                          MAINLINE LOGIC LOOP
000131 *
000132 ********************************************************************
000133          SPACE
000134 @MAIN    DS    0H
000135          SPACE
000136          ST    R14,W@MAIN               SAVE RETURN ADDR
000137          SPACE
000138          WTO   'HELLO, WORLD!'          MESSAGE TO OPERATOR
000139          SPACE
000140          L     R14,W@MAIN               LOAD RETURN ADDR
000141          BR    R14                      RETURN TO CALLER
000142          EJECT
000143 ********************************************************************
000144 *
000145 *        @FINAL                         FINALIZE PROGRAM
000146 *
000147 ********************************************************************
000148          SPACE
000149 @FINAL   DS    0H
000150          SPACE
000151          ST    R14,W@FINAL              SAVE RETURN ADDR
000152          SPACE
000153          SPACE
000154          L     R14,W@FINAL              LOAD RETURN ADDR
000155          BR    R14                      RETURN TO CALLER
000156          EJECT
```

```
000157 ************************************************************************
000158 *
000159 *        W O R K I N G   S T O R A G E
000160 *
000161 ************************************************************************
000162          SPACE
000163 *                                       ******************************
000164 *                                       *  EQUATES
000165 *                                       ******************************
000166          SPACE
000167 ERNOMEM  EQU   4                        GETMAIN FAILED
000168          EJECT
000169 ************************************************************************
000170 *
000171 *        $DATA                          DATA SECTION
000172 *
000173 ************************************************************************
000174          SPACE
000175 $DATA    DSECT
000176          SPACE
000177 *                                       ******************************
000178 *                                       *  REGISTER SAVE AREAS
000179 *                                       ******************************
000180          SPACE
000181          DS    18F                      REG SAVE AREA
000182          SPACE
000183 W@INIT   DS    F                        @INIT RETURN ADDR
000184 W@MAIN   DS    F                        @MAIN RETURN ADDR
000185 W@FINAL  DS    F                        @FINAL RETURN ADDR
000186          SPACE
000187 E$DATAL  EQU   *-$DATA                  LENGTH OF SAVE AREA
000188          EJECT
000189          END   $CODE
```

Sample program ASM003 introduces the allocation and deallocation of storage using the GETMAIN and FREEMAIN macros. ASM003 also introduces the $DATA data section (DSECT) and three subroutines, @INIT, @MAIN and @FINAL.

ASM003 - Job Control

The Job Control stream for ASM003 is the same as that for ASM002 except that references to member ASM002 have been changed to ASM003.

ASM003 - System Log

The System Log for ASM003 is the same as that for ASM002.

ASM003 - Assembler Listing

```
                                        EXTERNAL SYMBOL DICTIONARY                                      PAGE    1

SYMBOL   TYPE  ID   ADDR  LENGTH LDID                                                    ASM 0201 14.51 05/17/10

          PC  0001 000000 000000
$CODE     SD  0002 000000 00009C
                                                                                                      PAGE    2

  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10
                                      1 ********************************************************************
                                      2 *
                                      3 *        PROGRAM    ASM003
                                      4 *
                                      5 *        SAMPLE ASSEMBLY-LANGUAGE PROGRAM ILLUSTRATING THE USE OF
                                      6 *        THE GETMAIN AND FREEMAIN MACROS. THIS PROGRAM BUILDS ON
                                      7 *        THE SAMPLE PROGRAM ASM002, ADDING LOGIC TO ALLOCATE LOCAL
                                      8 *        STORAGE AND CALL SUBROUTINES.
                                      9 *
                                     10 *        ENTRY       1     PROGRAM ARGUMENTS ADDRESS
                                     11 *                    13     CALLER SAVE AREA ADDRESS
                                     12 *                    14     RETURN ADDRESS
                                     13 *                    15     CONTROL SECTION ADDRESS
                                     14 *
                                     15 *        EXIT        15     RETURN CODE
                                     16 *
                                     17 ********************************************************************
```

```
                                                                      PAGE    3

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

                                   19          COPY  MEQUREGS

                         00000     21 R0       EQU   0                       SYSTEM/TEMP USE
                         00001     22 R1       EQU   1                       SYSTEM/TEMP USE
                         00002     23 R2       EQU   2                       VOLATILE/PARAMETER
                         00003     24 R3       EQU   3                       VOLATILE/PARAMETER
                         00004     25 R4       EQU   4                       NON-VOLATILE
                         00005     26 R5       EQU   5                       NON-VOLATILE
                         00006     27 R6       EQU   6                       NON-VOLATILE
                         00007     28 R7       EQU   7                       NON-VOLATILE
                         00008     29 R8       EQU   8                       NON-VOLATILE
                         00009     30 R9       EQU   9                       NON-VOLATILE
                         0000A     31 R10      EQU   10                      CSECT BASE ADDR 2
                         0000B     32 R11      EQU   11                      CSECT BASE ADDR 1
                         0000C     33 R12      EQU   12                      DSECT BASE ADDR
                         0000D     34 R13      EQU   13                      SAVE AREA ADDR
                         0000E     35 R14      EQU   14                      LINKAGE/RETURN ADDR
                         0000F     36 R15      EQU   15                      LINKAGE/ENTRY ADDR/RESULT CODE
                                                                      PAGE    4

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

                                   39 ********************************************************************
                                   40 *
                                   41 *        USINGS
                                   42 *
                                   43 ********************************************************************

                         00000     45          USING $CODE,R11,R10           RESOLVE CODE TO REG 11,10
                         00000     46          USING $DATA,R13               RESOLVE DATA TO REG 13
```

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10
                                     48 ************************************************************************
                                     49 *
                                     50 *        $CODE                          PROGRAM CODE SECTION
                                     51 *
                                     52 ************************************************************************

000000                               54 $CODE    CSECT                          PROGRAM ENTRY

                                     56 *                                       ******************************
                                     57 *                                       *   SAVE CALLER REGS
                                     58 *                                       *   SETUP BASE REGS
                                     59 *                                       *   ALLOCATE STORAGE
                                     60 *                                       ******************************

000000 90EC D00C      0000C          62          STM   R14,R12,12(R13)          SAVE CALLER REG 14-15,0-12
000004 18BF                          63          LR    R11,R15                  LOAD 1ST CSECT BASE ADDR
000006 41F0 0001      00001          64          LA    R15,1                    ONE
00000A 41AF BFFF      00FFF          65          LA    R10,4095(R15,R11)        LOAD 2ND CSECT BASE ADDR

00000E 4100 0054      00054          67          LA    R0,E$DATAL               LENGTH OF STORAGE

                                     69          GETMAIN R,LV=(0)               ALLOCATE STORAGE
                                     70+*      OS/VS2 RELEASE 4 VERSION -- 10/21/75                              00004804
000012 4510 B016      00016          71+         BAL   1,*+4                              INDICATE GETMAIN       00682402
000016 0A0A                          72+         SVC   10                                 ISSUE GETMAIN SVC      00820002

000018 41F0 0004      00004          74          LA    R15,ERNOMEM              ASSUME GETMAIN ERROR
00001C 1211                          75          LTR   R1,R1                    IF MEMORY ALLOCATED
00001E 4780 B05C      0005C          76          BZ    @CODE20

                                     78 *                                         ****************************
                                     79 *                                         *  RESET STORAGE TO NULLS
                                     80 *                                         *  SETUP STORAGE BASE ADDR
                                     81 *                                         ****************************

000022 1821                          83          LR    R2,R1                      TARGET ADDR IS STORAGE
000024 4130 0054      00054          84          LA    R3,E$DATAL                 LENGTH IS STORAGE LENGTH
000028 1744                          85          XR    R4,R4                      SOURCE ADDR IS NULL
00002A 1854                          86          LR    R5,R4                      SOURCE LENGTH IS ZERO (PAD)
00002C 0E24                          87          MVCL  R2,R4                      ZERO STORAGE
```

```
00002E 50D1 0004      00004          89          ST    R13,4(R1)               SAVE CALLER STORAGE ADDR
000032 5011 0008      00008          90          ST    R1,8(R1)                SAVE OUR STORAGE ADDR
000036 18D1                          91          LR    R13,R1                  LOAD STORAGE BASE REG

                                     93 *                                      ****************************
                                     94 *                                      *  DO PROGRAM INITIALIZATION
                                     95 *                                      *  DO PROGRAM MAINLINE
                                     96 *                                      *  DO PROGRAM FINALIZATION
                                     97 *                                      ****************************

000038 17FF                          99          XR    R15,R15                 ZERO RETURN CODE

00003A 45E0 B066      00066         101          BAL   R14,@INIT               DO INITIALIZATION
                                                                                                    PAGE    6

  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

00003E 12FF                         103          LTR   R15,R15                 IF INITIALIZATION OK
000040 4770 B048      00048         104          BNZ   @CODE10

000044 45E0 B070      00070         106          BAL   R14,@MAIN                 DO MAINLINE

000048                              108 @CODE10  DS    0H                      END IF

000048 45E0 B092      00092         110          BAL   R14,@FINAL              DO FINALIZATION

                                    112 *                                      ****************************
                                    113 *                                      *  RESTORE CALLER SAVE AREA
                                    114 *                                      *  RELEASE STORAGE
                                    115 *                                      ****************************

00004C 181D                         117          LR    R1,R13                  OUR SAVE AREA ADDR
00004E 58D1 0004      00004         118          L     R13,4(R1)               CALLER SAVE AREA ADDR

000052 4100 0054      00054         120          LA    R0,E$DATAL              LENGTH OF SAVE AREA

                                    122          FREEMAIN R,LV=(0),A=(R1)      RELEASE STORAGE
                                   123+*      OS/VS2 RELEASE 3 VERSION  -- 10/25/74                          00001603
000056 4110 1000      00000        124+         LA    1,0(0,R1)                     LOAD AREA ADDRESS      00164002
00005A 0A0A                        125+         SVC   10                            ISSUE FREEMAIN SVC     00311202

00005C                             127 @CODE20  DS    0H                   END IF
```

```
                                  129 *                                  ****************************
                                  130 *                                  *  RESTORE CALLER REGS
                                  131 *                                  *  RETURN TO CALLER
                                  132 *                                  ****************************

00005C 980C D014      00014       134          LM    R0,R12,20(R13)      RESTORE CALLER REG 0-12
000060 58ED 000C      0000C       135          L     R14,12(R13)         RESTORE CALLER REG 14
000064 07FE                       136          BR    R14                 RETURN TO CALLER
                                                                                                PAGE    7

  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

                                  138 ************************************************************************
                                  139 *
                                  140 *        @INIT                       INTIIALIZE PROGRAM
                                  141 *
                                  142 ************************************************************************

000066                            144 @INIT    DS    0H

000066 50E0 D048      00048       146          ST    R14,W@INIT          SAVE RETURN ADDR


00006A 58E0 D048      00048       149          L     R14,W@INIT          LOAD RETURN ADDR
00006E 07FE                       150          BR    R14                 RETURN TO CALLER
```

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                    ASM 0201 14.51 05/17/10

                                    152 ********************************************************************
                                    153 *
                                    154 *        @MAIN                          MAINLINE LOGIC LOOP
                                    155 *
                                    156 ********************************************************************

000070                              158 @MAIN    DS    0H

000070 50E0 D04C      0004C         160          ST    R14,W@MAIN               SAVE RETURN ADDR

                                    162          WTO   'HELLO, WORLD!'          MESSAGE TO OPERATOR
000074                              163+         CNOP  0,4                                                 09800002
000074 4510 B08A      0008A         164+         BAL   1,IHB0003A                        BRANCH AROUND MESSAGE  09850002
000078 0011                         165+         DC    AL2(17)             TEXT LENGTH                     13200002
00007A 0000                         166+         DC    B'0000000000000000' MCS FLAGS                       13250002
00007C C8C5D3D3D66B40E6             167+         DC    C'HELLO, WORLD!'                                    13350002
00008A                              168+IHB0003A DS    0H                                                  15850002
00008A 0A23                         169+         SVC   35                                                  15950002

00008C 58E0 D04C      0004C         171          L     R14,W@MAIN               LOAD RETURN ADDR
000090 07FE                         172          BR    R14                      RETURN TO CALLER
```



```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                    ASM 0201 14.51 05/17/10

                                    174 ********************************************************************
                                    175 *
                                    176 *        @FINAL                         FINALIZE PROGRAM
                                    177 *
                                    178 ********************************************************************

000092                              180 @FINAL   DS    0H

000092 50E0 D050      00050         182          ST    R14,W@FINAL              SAVE RETURN ADDR


000096 58E0 D050      00050         185          L     R14,W@FINAL              LOAD RETURN ADDR
00009A 07FE                         186          BR    R14                      RETURN TO CALLER
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

                                   188 ************************************************************************
                                   189 *
                                   190 *        W O R K I N G   S T O R A G E
                                   191 *
                                   192 ************************************************************************

                                   194 *                                    ******************************
                                   195 *                                    *  EQUATES
                                   196 *                                    ******************************

                         00004     198 ERNOMEM  EQU   4                     GETMAIN FAILED
```



```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

                                   200 ************************************************************************
                                   201 *
                                   202 *        $DATA                       DATA SECTION
                                   203 *
                                   204 ************************************************************************

000000                             206 $DATA    DSECT

                                   208 *                                    ******************************
                                   209 *                                    *  REGISTER SAVE AREAS
                                   210 *                                    ******************************

000000                             212          DS    18F                   REG SAVE AREA

000048                             214 W@INIT   DS    F                     @INIT RETURN ADDR
00004C                             215 W@MAIN   DS    F                     @MAIN RETURN ADDR
000050                             216 W@FINAL  DS    F                     @FINAL RETURN ADDR

                         00054     218 E$DATAL  EQU   *-$DATA               LENGTH OF SAVE AREA
```



```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                ASM 0201 14.51 05/17/10

000000                             220          END   $CODE
```

```
SYMBOL    LEN   VALUE   DEFN    REFERENCES                                                    ASM 0201 14.51 05/17/10

$CODE     00001 00000000 00054  00045 00220
$DATA     00001 00000000 00206  00046 00218
@CODE10   00002 00000048 00108  00104
@CODE20   00002 0000005C 00127  00076
@FINAL    00002 00000092 00180  00110
@INIT     00002 00000066 00144  00101
@MAIN     00002 00000070 00158  00106
E$DATAL   00001 00000054 00218  00067 00084 00120
ERNOMEM   00001 00000004 00198  00074
IHB0003A  00002 0000008A 00168  00164
R0        00001 00000000 00021  00067 00120 00134
R1        00001 00000001 00022  00075 00075 00083 00089 00090 00090 00091 00117 00118 00124
R10       00001 0000000A 00031  00045 00065
R11       00001 0000000B 00032  00045 00063 00065
R12       00001 0000000C 00033  00062 00134
R13       00001 0000000D 00034  00046 00062 00089 00091 00117 00118 00134 00135
R14       00001 0000000E 00035  00062 00101 00106 00110 00135 00136 00146 00149 00150 00160 00171 00172 00182 00185 00186
R15       00001 0000000F 00036  00063 00064 00065 00074 00099 00099 00103 00103
R2        00001 00000002 00023  00083 00087
R3        00001 00000003 00024  00084
R4        00001 00000004 00025  00085 00085 00086 00087
R5        00001 00000005 00026  00086
R6        00001 00000006 00027
R7        00001 00000007 00028
R8        00001 00000008 00029
R9        00001 00000009 00030
W@FINAL   00004 00000050 00216  00182 00185
W@INIT    00004 00000048 00214  00146 00149
W@MAIN    00004 0000004C 00215  00160 00171
```

```
NO STATEMENTS FLAGGED IN THIS ASSEMBLY
HIGHEST SEVERITY WAS    0
OPTIONS FOR THIS ASSEMBLY
  ALIGN, ALOGIC, BUFSIZE(STD), DECK, ESD, FLAG(0), LINECOUNT(55), LIST, NOMCALL, YFLAG, WORKSIZE(2097152)
  NOMLOGIC, NONUMBER, OBJECT, RENT, RLD, NOSTMT, NOLIBMAC, NOTERMINAL, NOTEST, XREF
  SYSPARM()
WORK FILE BUFFER SIZE/NUMBER =13030/ 1
TOTAL RECORDS READ FROM SYSTEM INPUT        189
TOTAL RECORDS READ FROM SYSTEM LIBRARY     1846
TOTAL RECORDS PUNCHED                        6
TOTAL RECORDS PRINTED                      281

 F64-LEVEL LINKAGE EDITOR OPTIONS SPECIFIED XREF,LET,LIST,NCAL,RENT
          DEFAULT OPTION(S) USED -  SIZE=(165888,55296)
```

CROSS REFERENCE TABLE

```
  CONTROL SECTION                       ENTRY

    NAME    ORIGIN  LENGTH                NAME   LOCATION     NAME   LOCATION     NAME   LOCATION     NAME   LOCATION
  $CODE         00      9C


 ENTRY ADDRESS       00

 TOTAL LENGTH        A0
****ASM003    NOW REPLACED IN DATA SET
AUTHORIZATION CODE IS         0.
**MODULE HAS BEEN MARKED REENTERABLE, AND REUSABLE.
```

ASM003 - Notes

On line 45 of the assembler listing, we have added a second base register to the USING statement for the $CODE control section. Our control section is still very small, but we will need an additional base register if our control section size exceeds 4096 bytes. On lines 64 and 65 we load the second base register, register 10, with the contents of our first base register, register 11, plus 4096. To do this efficiently wih the LA instruction, we add a maximum displacement of 4095 plus a temporary index value of 1 to the value in register 11 and place that sum in register 10. The assembler will know to use register 10 in all base-displacement codings for symbols in the second 4KB of the control section.

Over time, as a program grows larger, using a single contiguous control section will lead to more and more registers needing to be reserved as base registers. This is not practical for large programs that need to efficiently implement complex algorithms because such implementations also consume general registers, albeit temorarily. We will introduce a technique in later examples for minimizing the consumption of registers as base registers.

After setting up the base registers, ASM003 allocates storage for use storing variables such as return addresses. This allocation is accomplished by a supervisor interrupt, generated by the GETMAIN macro. In this example, on line 69, GETMAIN requires only that the length of the storage has been placed in register 0. On return from the supervisor interrupt, register 1 contains the address of the allocated storage or zero if no storage could be allocated.

After issuing the GETMAIN, ASM003 enters a conditional "IF" statement section of code if R1 is non-zero on return from the GETMAIN. If storage has been allocated, it is reset to nulls using the MVCL command. Then, we save the addresses of our caller's storage area and our own storage area in the allocated storage. Register 13 is then loaded with the address of our allocated storage. Note that in the USINGS section, on line 46, the assembler is directed to resolve base-displacement addresses for symbols found in the $DATA data section to register 13.

Next, we reset register 15 to zero. ASM003 will use register 15 to hold result codes from subroutines and from the entire program. A call is made to the @INIT subroutine using the branch-and-link instruction (BAL). Register 14 is generally used to hold the return address. If R15 is zero upon return from @INIT, then @MAIN and @FINAL will be called. Otherwise, only @FINAL is called.

After returning from @FINAL, ASM003 restores register 13 to its original value, pointing to the caller's save area. Then, the FREEMAIN macro is used to issue a supervisor call to release the storage allocated earlier by the GETMAIN macro. In order to make sure we release the correct storage, we pass the address of the storage to FREEMAIN in register 1, and the length of the storage in register 0. ASM003 returns to its caller in the same manner as ASM002 - by restoring the general registers to their original state, with the exception of register 15, our result code.

The initialization function, @INIT, and the finalization function, @FINAL, do nothing in ASM003 except save and restore the return address found in register 14. Special save areas have been defined in the $DATA data section to hold return addresses from each subroutine. This is effective provided that no subroutine is recursive, as this would overwrite the subroutine's return address. The mainline function, @MAIN, issues our original WTO macro to send a message to the operator console.

The "Working Storage" section will include all symbolic constants and variable data areas. Constants and tables will actually be defined as part of our $CODE control section. Variable data will be defined in a $DATA data section, which maps the storage allocated by the GETMAIN call.

ASM003 provides a skeleton for just about any S/370 assembler program. Generally, in the @INIT function, a program will allocate additional storage areas, open files, do any initial reads of data, etc. In the @FINAL subroutine, the program will close files and release allocated storage other than $DATA itself. In the @MAIN subroutine, the program's main logic "loop" is defined - either processing records in batch until and end-of-file condition is reached, or handing other input and output from and to devicies until the program is otherwise terminated.