TPF Users Group
Grapevine, Texas

IBM Software Group

GNU Compiler Collection (GCC)

Colette A. Manoni
October 2004

AIM Core and Enterprise Solutions
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

What is the GNU Compiler Collection?

- A set of programming language compilers produced by the GNU Project
- Started with support for C Language, and added the following:
  - Ada
  - C++
  - Fortran
  - Java™
  - Objective C
- TPF only supports C/C++

Why use GCC?

- Widely available
  - Supports large number of architectures and operating systems
- Open community
  - Large number of mailing lists
  - Dedicated core of developers
- Code is portable
- Large number of tools available
- Can be used as a cross compiler
  - Compiler does not need to run on target platform

GCC for z/TPF

- TPF is a supported platform
  - TPF platform code under S/390® tree
  - GCC built as cross compiler for z/TPF
    - Minimum stack frame size is larger
      - Needed to support integration of C/C++ with TPF Assembler
  - Machine dependent option -mtpf-trace
    - Prolog / Epilog Hooks
      - Needed to support z/TPF trace tools

Code Page Support

- GCC by default generates character literals in ascii
- Long standing requirement to support additional code pages
- TPF organization helped to drive this support due to the need for IBM1047
- Implementation is based on iconv()
  - Any supported code page can be specified
  - example: -fexec-charset=IBM1047
- General option is available to all platforms

The -W option : Warning Message Generation

- Some examples:
  - -Wuninitialized
    - An automatic variable is used without first being initialized
  - -Wpadded
    - Padding is included in a structure, either to align an element of the structure or to align the whole structure.
  - -Wreturn-type
    - A function is defined with a return-type that defaults to int or a return statement with no return-value in a function whose return-type is not void.
- Function comparable to LINT tools

Other Compile Options

- Creating Shared Objects
  - Option needed is -fpic or -fPIC
  - Program produced is relocatable
  - Two flavors based on size of program
    - Heavy use of relative instructions
- Producing Debug Information
  - Option needed is -gdwarf-2
  - Information needed for the following functions/capabilities:
    - source statements in the listing
    - function information for zdmap
    - parameter information for stack and trace formatting
    - symbol information for debugger

GCC Extenstion: Attributes

- Keyword __attribute__ used to alter default characteristics
  - Functions
    - noreturn, noinline, always_inline
  - Types
    - aligned, packed, transparent_union, unused, deprecated
  - Variables
    - aligned, mode, cleanup

- Attribute mechanism used to create 32-bit pointer
  - mode defines the size of the variable
  - SI is single integer (4 bytes)
  - void * ptr __attribute__ ((mode (SI)))
    - prt is allocated and accessed as 4-bytes
  - size_t size_t32 __attribute__ ((mode (SI)))
    - size_t32 is allocated and accessed as 4-bytes

GCC Extension: Embedded Assembler

```
static inline int  __attribute__ ((unused))
cs(cs_t *oldptr, cs_t *curptr, cs_t newvalue)  {

 int rc;

 __asm__ __volatile__ (
       "                                \n"
       "        l     %%r0,%1           \n"    /* load *oldptr into r0        */
       "        cs    %%r0,%3,%2        \n"    /* execute cs                  */
       "        jz    0f                \n"    /* if cs is successful, branch */
       "        st    %%r0,%1           \n"    /* cs failed, store r0 into *oldptr */
       "        lghi  %0,1              \n"    /* load failure return code    */
       "        j     1f                \n"    /* branch to routine exit      */
       "0:                              \n"
       "        lghi  %0,0              \n"    /* load success return code    */
       "1:                              \n"
       : "=r" (rc), "=m" (*oldptr), "=Q" (*curptr)
       : "d" (newvalue), "m" (*oldptr), "Q" (*curptr)
       : "0", "cc", "memory");

 return rc;
```

Function Call Linkage

- C/C++ parameter format
  - Same for both 'C' and 'C++'
  - R2 - R6 used for passing the first 5 parameters.
    - Additional parameters are stored in the calling functions stack frame
  - See ABI document for more details
- TPF assembler calling C
  - New macros provided: CPROC and CALLC
  - Details covered in single source presentation
- Interface for calling BAL segments from C functions
  - TPF_regs
  - Behaves the same as today - default is 31-bit mode
  - New macros provided for calling 64-bit assembler

Register Conventions

r0, r1	General purpose	Volatile[1]
r2	Parameter passing and return values	Volatile
r3, r4, r5	Parameter passing	Volatile
r6	Parameter passing	Saved[2]
r7 - r11	Local variables	Saved
r12	Local variable, commonly used as GOT pointer	Saved
r13	Local variable, commonly used as Literal Pool pointer	Saved
r14	Return address	Volatile
r15	Stack pointer	Saved
f0, f2, f4, f6	Parameter passing and return values	Volatile
f1, f3, f5, f7	General purpose	Volatile
f8 –f15	General purpose	Saved
Access registers 0,1	Reserved for system use	Volatile
Access registers 2-15	General purpose	Volatile

[1]Volatile: These registers are not preserved across function calls.

[2]Saved: These registers belong to the calling function. A called function shall save these registers' values before it changes them, restoring their values before it returns.

From the zSeries ELF Application Binary Interface Supplement v.02 Edition, Published 18 November 2002

Binary Utilities (binutils)

- c++filt
  - takes mangled C++ name and translates back to original
- nm
  - lists the symbols defined in an object file
- objdump
  - displays different information from one or more object files
- readelf :
  - displays information about ELF files
- size
  - lists the name and size of each section in an object file

Sample Program

```
#include <stdio.h>
#include <stdlib.h>

extern int average;                                    /* external data */
int get_average(int num1, int num2, int num3);         /* external function */

int QZZ2()   {

        int num1 = 1;
        int num2 = 2;
        int num3 = 3;

        average =  get_average(num1, num2, num3);
        printf("average is %d\n", average);

    return 0;
}
```

Sample Listing - compile options

```
 1 .file "file2.c"
 2 # GNU C version 3.4-gnupro-04r1-3 (s390x-ibm-tpf)
 3 # compiled by GNU C version 3.2.2.
 4 # GGC heuristics: --param ggc-min-expand=43 --param ggc-min-heapsize=25000
 5 # options passed:  -I- -I/ztpf/cur/local_mod/base/include
 6 # -I/ztpf/cur/base/include -I/ztpf/cur/local_mod/opensource/include
 7 # -I/ztpf/cur/opensource/include -I/ztpf/cur/tpfdf/include -D_GNU_SOURCE
 8 # -D_TPF_SOURCE -m64 -mzarch -march=z900 -auxbase -gdwarf-2 -O3 -Wall
 9 # -Wno-format-extra-args -fpic -fexec-charset=IBM1047 -fverbose-asm
10 # -fmessage-length=0 -funsigned-char -fno-builtin-abort -fno-builtin-exit
11 # options enabled:  -feliminate-unused-debug-types -fdefer-pop
12 # -fomit-frame-pointer -foptimize-sibling-calls -funit-at-a-time
13 # -fcse-follow-jumps -fcse-skip-blocks -fexpensive-optimizations
14 # -fthread-jumps -fstrength-reduce -funswitch-loops -fpeephole -fforce-mem
 ......
```

Sample Listing - read only data

```
35                .Ltext0:
36                 .section .rodata.str1.2,"aMS",@progbits,1
37                 .align 2
38                .LC0:
39 0000 81A58599  .string "\201\245\205\231\201\207\205@\211\242@l\204\025"
39      81878540
39      89A2406C
39      841500
```

Sample Listing - prologue

```
40 000f 00          .text
41                  .align 4
42                 .globl QZZ2
44                 QZZ2:
45                 .LFB30:
46                  .file 1 "/home/C/base/rt/file2.c"
 1:/home/C/base/rt/file2.c **** #include <stdio.h>
 2:/home/C/base/rt/file2.c **** #include <stdlib.h>
 3:/home/C/base/rt/file2.c ****
 4:/home/C/base/rt/file2.c **** extern int average;
 5:/home/C/base/rt/file2.c **** int get_average(int num1, int num2, int num3);
 6:/home/C/base/rt/file2.c ****
 7:/home/C/base/rt/file2.c **** int QZZ2()
 8:/home/C/base/rt/file2.c **** {
47                  .loc 1 8 0
48 0000 EBCFF060    stmg %r12,%r15,96(%r15) #,,0024
49                 .LCFI0:
50 0006 B904002F    lgr %r2,%r15 #,
51 000a A7FBFE40    aghi %r15,-448 #,
52                 .LCFI1:
53 000e C0C00000    larl %r12,_GLOBAL_OFFSET_TABLE_ #,0000
54 0014 E320F000    stg %r2,0(%r15) #,0024
55 001a 4D100FE0    bas %r1,4064
```

Sample Listing - function calls

```
9:/home/C/base/rt/file2.c ****
10:/home/C/base/rt/file2.c **** int num1 = 1;
11:/home/C/base/rt/file2.c **** int num2 = 2;
12:/home/C/base/rt/file2.c **** int num3 = 3;
13:/home/C/base/rt/file2.c ****
14:/home/C/base/rt/file2.c **** average =  get_average(num1, num2, num3);

56                 .loc 1 14 0
57 001e A7490003   lghi %r4,3 # num3,
58 0022 A7390002   lghi %r3,2 # num2,
59 0026 A7290001   lghi %r2,1 # num1,
60 002a C0E50000   brasl %r14,get_average@PLT #,0000
61 0030 E310C000   lg %r1,average@GOT(%r12) # tmp48,0004
62 0036 B9040002   lgr %r0,%r2 # tmp47,

15:/home/C/base/rt/file2.c ****
16:/home/C/base/rt/file2.c **** printf("average is %d\n", average);

63                 .loc 1 16 0
64 003a B9040032   lgr %r3,%r2 # average, tmp47
65                 .loc 1 14 0
66 003e 50001000   st %r0,0(%r1) # tmp47, average
67                 .loc 1 16 0
68 0042 C0200000   larl %r2,.LC0 #,0000
69 0048 C0E50000   brasl %r14,printf@PLT #,0000
```

Sample Listing - eplilog

```
17:/home/C/base/rt/file2.c ****
18:/homeC/base/rt/file2.c **** return 0;
19:/home/C/base/rt/file2.c ****
20:/home/C/base/rt/file2.c **** }

70                .loc 1 20 0
71 004e A7290000   lghi %r2,0 # <result>,
72 0052 4D100FE6   bas %r1,4070
73 0056 E340F230   lg %r4,560(%r15) #,
73      0004
74 005c EBCFF220   lmg %r12,%r15,544(%r15) #,,
74      0004
75 0062 07F4       br %r4 #
```

Sample Listing - debug information

```
116                  .Lframe1:
117 0000 00000014      .4byte .LECIE1-.LSCIE1
118                  .LSCIE1:
119 0004 00000000      .4byte 0x0
120 0008 01            .byte 0x1
121 0009 00            .string ""
122 000a 01            .uleb128 0x1
123 000b 78            .sleb128 -8
124 000c 0E            .byte 0xe

   ....
```

Sample Listing - symbols

```
DEFINED SYMBOLS
                            *ABS*:0000000000000000 file2.c
    /tmp/ccCXye4G.s:44     .text:0000000000000000 QZZ2
                  .rodata.str1.2:0000000000000000 .LC0
                      .debug_str:0000000000000000 .LASF14
                      .debug_str:0000000000000053 .LASF15
                      .debug_str:0000000000000018 .LASF0
                      .debug_str:0000000000000080 .LASF1

                          .....


UNDEFINED SYMBOLS
_GLOBAL_OFFSET_TABLE_
get_average
average
printf
```

Sample Output - nm

```
nm -S QZZ2.so

0000000000000808 0000000000000064 T QZZ2
0000000000003000 A _DYNAMIC
0000000000002000 A _GLOBAL_OFFSET_TABLE_
0000123a234535c8 a __TPF_TOD
0000000000003280 A __bss_start
0000000000003280 A _edata
0000000000003280 A _end
                 U average
                 U find_average
00000000000007a0 0000000000000066 T get_average
                 U printf
```

objdump Output

```
0000000000000808 <QZZ2>:
 808:    eb cf f0 60 00 24 stmg %r12,%r15,96(%r15)
 80e:    b9 04 00 2f       lgr  %r2,%r15
 812:    a7 fb fe 40       aghi %r15,-448
 816:    c0 c0 00 00 0b f5 larl %r12,2000 <_GLOBAL_OFFSET_TABLE_>
 81c:    e3 20 f0 00 00 24 stg  %r2,0(%r15)
 822:    4d 10 0f e0       bas  %r1,4064
 826:    a7 49 00 03       lghi %r4,3
 82a:    a7 39 00 02       lghi %r3,2
 82e:    a7 29 00 01       lghi %r2,1
 832:    c0 e5 ff ff ff a7 brasl    %r14,780 <get_average-0x20>
 838:    e3 10 c0 30 00 04 lg   %r1,48(%r12)
 83e:    b9 04 00 02       lgr  %r0,%r2
 842:    b9 04 00 32       lgr  %r3,%r2
 846:    50 00 10 00       st   %r0,0(%r1)
 84a:    c0 20 00 00 00 19 larl %r2,87c <QZZ2+0x74>
 850:    c0 e5 ff ff ff 78 brasl    %r14,740 <get_average-0x60>
 856:    a7 29 00 00       lghi %r2,0
 85a:    4d 10 0f e6       bas  %r1,4070
 85e:    e3 40 f2 30 00 04 lg   %r4,560(%r15)
 864:    eb cf f2 20 00 04 lmg  %r12,%r15,544(%r15)
 86a:    07 f4             br   %r4
```

prolog

get_average()

printf()

epilog

readelf Output

```
Magic:   7f 45 4c 46 02 02 01 00 00 00 00 00 00 00 00 00
Class:                             ELF64
Data:                              2's complement, big endian
Version:                           1 (current)
OS/ABI:                            UNIX - System V
ABI Version:                       0
Type:                              DYN (Shared object file)
Machine:                           IBM S/390
Version:                           0x1
Entry point address:               0x808
...
```

readelf Output

```
Dynamic segment at offset 0x3000 contains 30 entries:
  Tag        Type                         Name/Value
 0x0000000000000001 (NEEDED)             Shared library: [QZZ3]
 0x0000000000000001 (NEEDED)             Shared library: [CTIS]
 0x0000000000000001 (NEEDED)             Shared library: [CISO]
 0x0000000000000001 (NEEDED)             Shared library: [CLBM]
 0x0000000000000001 (NEEDED)             Shared library: [CTAL]
 0x0000000000000001 (NEEDED)             Shared library: [CFVS]
 0x0000000000000001 (NEEDED)             Shared library: [CTBX]
 0x0000000000000001 (NEEDED)             Shared library: [CTXO]
 0x0000000000000001 (NEEDED)             Shared library: [CJ00]
 0x0000000000000001 (NEEDED)             Shared library: [CTDF]
 0x0000000000000001 (NEEDED)             Shared library: [COMX]
 0x0000000000000001 (NEEDED)             Shared library: [COMS]
 0x0000000000000001 (NEEDED)             Shared library: [CTHD]
 0x0000000000000001 (NEEDED)             Shared library: [CTAD]
 0x0000000000000001 (NEEDED)             Shared library: [TPFSTUB]
 0x000000000000000e (SONAME)             Library soname: [QZZ2]
```

readelf Output

```
Symbol table '.dynsym' contains 35 entries:
   Num:    Value          Size Type    Bind   Vis      Ndx Name
     0: 0000000000000000     0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000000000e8     0 SECTION LOCAL  DEFAULT    1
     ...
    25: 0000000000000000   222 FUNC    GLOBAL DEFAULT  UND printf
    26: 0000000000003000     0 OBJECT  GLOBAL DEFAULT  ABS _DYNAMIC
    27: 0000000000000808   100 FUNC    GLOBAL DEFAULT    8 QZZ2
    28: 0000000000000000     4 OBJECT  GLOBAL DEFAULT  UND average
    29: 0000000000000000    70 FUNC    GLOBAL DEFAULT  UND find_average
    30: 0000000000003280     0 NOTYPE  GLOBAL DEFAULT  ABS __bss_start
    31: 00000000000007a0   102 FUNC    GLOBAL DEFAULT    8 get_average
    ...
```

Sample Output - size

```
size QZZ2.so

   text     data      bss      dec      hex filename
   1950      808        0     2758      ac6 QZZ2.so
```

References

- http://www.gnu.org
  - Information about FSF
  - Links to open source packages
  -
- http://gcc.gnu.org
  - Manuals
  - Mailing Lists
  -
- http://www.google.com
  - Search for more information !

Questions?

Legal

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

The following are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

- S/390
- zSeries

Other company, product and service names may be trademarks or service marks of others