0. Assignment Info

- Author : 2019320090 컴퓨터학과 강지원
- Assignment #1

1. Explanation of the role of Makefile in Eclipse project

'Makefile' contains the information about 'compliation steps' (preprocessing, compilation, assembling, linking etc) for the target files specified in 'Makefile'. Also, it contains some configurations needed for compilation. For example, path and configurations of compile, assembler, linker are included in 'Makefile'.

2. Outputs of each compilation step in the native compilation

Preprocessor : code1.i

```
# 0 "code1.c"
# 0 "<built-in>"
# 0 "<command-line>"
# 1 "code1.c"

int compare(int b, int c)
{
    int a;
    a = ((b) < (c) ? (b) : (c));;
    return a;
}
```

Compilation : code1.s

```
    .file   "code1.c"
    .text
.Ltext0:
    .file 0 "/home/jiwon/workspace/x86" "code1.c"
    .globl  compare
    .def    compare;    .scl    2;  .type   32; .endef
    .seh_proc   compare
compare:
.LFB0:
    .file 1 "code1.c"
    .loc 1 3 1
    pushq   %rbp
    .seh_pushreg    %rbp
```

```
.LCFI0:
    movq    %rsp, %rbp
    .seh_setframe   %rbp, 0
.LCFI1:
    subq    $16, %rsp
    .seh_stackalloc 16
    .seh_endprologue
    movl    %ecx, 16(%rbp)
    movl    %edx, 24(%rbp)
    .loc 1 5 7
    movl    24(%rbp), %edx
    movl    16(%rbp), %eax
    cmpl    %eax, %edx
    cmovle  %edx, %eax
    movl    %eax, -4(%rbp)
    .loc 1 6 12
    movl    -4(%rbp), %eax
    .loc 1 7 1
    addq    $16, %rsp
    popq    %rbp
.LCFI2:
    ret
.LFE0:
    .seh_endproc
    .section    .debug_frame,"dr"
.Lframe0:
    .long   .LECIE0-.LSCIE0
.LSCIE0:
    .long   0xffffffff
    .byte   0x3
    .ascii "\0"
    .uleb128 0x1
    .sleb128 -8
    .uleb128 0x10
    .byte   0xc
    .uleb128 0x7
    .uleb128 0x8
    .byte   0x90
    .uleb128 0x1
    .align 8
.LECIE0:
.LSFDE0:
    .long   .LEFDE0-.LASFDE0
.LASFDE0:
    .secrel32   .Lframe0
    .quad   .LFB0
    .quad   .LFE0-.LFB0
    .byte   0x4
    .long   .LCFI0-.LFB0
    .byte   0xe
    .uleb128 0x10
    .byte   0x86
    .uleb128 0x2
    .byte   0x4
```

```
    .long   .LCFI1-.LCFI0
    .byte   0xd
    .uleb128 0x6
    .byte   0x4
    .long   .LCFI2-.LCFI1
    .byte   0xc6
    .byte   0xc
    .uleb128 0x7
    .uleb128 0x8
    .align 8
.LEFDE0:
    .text
.Letext0:
    .section    .debug_info,"dr"
.Ldebug_info0:
    .long   0xa7
    .word   0x5
    .byte   0x1
    .byte   0x8
    .secrel32   .Ldebug_abbrev0
    .uleb128 0x2
    .ascii "GNU C17 11.3.0 -mtune=generic -march=x86-64 -g\0"
    .byte   0x1d
    .secrel32   .LASF0
    .secrel32   .LASF1
    .quad   .Ltext0
    .quad   .Letext0-.Ltext0
    .secrel32   .Ldebug_line0
    .uleb128 0x3
    .ascii "compare\0"
    .byte   0x1
    .byte   0x2
    .byte   0x5
    .long   0xa3
    .quad   .LFB0
    .quad   .LFE0-.LFB0
    .uleb128 0x1
    .byte   0x9c
    .long   0xa3
    .uleb128 0x1
    .ascii "b\0"
    .byte   0x11
    .long   0xa3
    .uleb128 0x2
    .byte   0x91
    .sleb128 0
    .uleb128 0x1
    .ascii "c\0"
    .byte   0x18
    .long   0xa3
    .uleb128 0x2
    .byte   0x91
    .sleb128 8
    .uleb128 0x4
```

```
        .ascii "a\0"
        .byte   0x1
        .byte   0x4
        .byte   0x9
        .long   0xa3
        .uleb128 0x2
        .byte   0x91
        .sleb128 -20
        .byte   0
        .uleb128 0x5
        .byte   0x4
        .byte   0x5
        .ascii "int\0"
        .byte   0
        .section    .debug_abbrev,"dr"
    .Ldebug_abbrev0:
        .uleb128 0x1
        .uleb128 0x5
        .byte   0
        .uleb128 0x3
        .uleb128 0x8
        .uleb128 0x3a
        .uleb128 0x21
        .sleb128 1
        .uleb128 0x3b
        .uleb128 0x21
        .sleb128 2
        .uleb128 0x39
        .uleb128 0xb
        .uleb128 0x49
        .uleb128 0x13
        .uleb128 0x2
        .uleb128 0x18
        .byte   0
        .byte   0
        .uleb128 0x2
        .uleb128 0x11
        .byte   0x1
        .uleb128 0x25
        .uleb128 0x8
        .uleb128 0x13
        .uleb128 0xb
        .uleb128 0x3
        .uleb128 0x1f
        .uleb128 0x1b
        .uleb128 0x1f
        .uleb128 0x11
        .uleb128 0x1
        .uleb128 0x12
        .uleb128 0x7
        .uleb128 0x10
        .uleb128 0x17
        .byte   0
        .byte   0
```

```
        .uleb128 0x3
        .uleb128 0x2e
        .byte   0x1
        .uleb128 0x3f
        .uleb128 0x19
        .uleb128 0x3
        .uleb128 0x8
        .uleb128 0x3a
        .uleb128 0xb
        .uleb128 0x3b
        .uleb128 0xb
        .uleb128 0x39
        .uleb128 0xb
        .uleb128 0x27
        .uleb128 0x19
        .uleb128 0x49
        .uleb128 0x13
        .uleb128 0x11
        .uleb128 0x1
        .uleb128 0x12
        .uleb128 0x7
        .uleb128 0x40
        .uleb128 0x18
        .uleb128 0x7a
        .uleb128 0x19
        .uleb128 0x1
        .uleb128 0x13
        .byte   0
        .byte   0
        .uleb128 0x4
        .uleb128 0x34
        .byte   0
        .uleb128 0x3
        .uleb128 0x8
        .uleb128 0x3a
        .uleb128 0xb
        .uleb128 0x3b
        .uleb128 0xb
        .uleb128 0x39
        .uleb128 0xb
        .uleb128 0x49
        .uleb128 0x13
        .uleb128 0x2
        .uleb128 0x18
        .byte   0
        .byte   0
        .uleb128 0x5
        .uleb128 0x24
        .byte   0
        .uleb128 0xb
        .uleb128 0xb
        .uleb128 0x3e
        .uleb128 0xb
        .uleb128 0x3
```

```
    .uleb128 0x8
    .byte   0
    .byte   0
    .byte   0
    .section    .debug_aranges,"dr"
    .long   0x2c
    .word   0x2
    .secrel32   .Ldebug_info0
    .byte   0x8
    .byte   0
    .word   0
    .word   0
    .quad   .Ltext0
    .quad   .Letext0-.Ltext0
    .quad   0
    .quad   0
    .section    .debug_line,"dr"
.Ldebug_line0:
    .section    .debug_str,"dr"
    .section    .debug_line_str,"dr"
.LASF1:
    .ascii "/home/jiwon/workspace/x86\0"
.LASF0:
    .ascii "code1.c\0"
    .ident  "GCC: (GNU) 11.3.0"
```

Assembler : code1.dump

```
code1.o:     file format pe-x86-64


Disassembly of section .text:

0000000000000000 <compare>:
#define min(x, y) ((x) < (y) ? (x) : (y));
int compare(int b, int c)
{
   0:   55                      push   rbp
   1:   48 89 e5                mov    rbp,rsp
   4:   48 83 ec 10             sub    rsp,0x10
   8:   89 4d 10                mov    DWORD PTR [rbp+0x10],ecx
   b:   89 55 18                mov    DWORD PTR [rbp+0x18],edx
    int a;
    a = min(b, c);
   e:   8b 55 18                mov    edx,DWORD PTR [rbp+0x18]
  11:   8b 45 10                mov    eax,DWORD PTR [rbp+0x10]
  14:   39 c2                   cmp    edx,eax
  16:   0f 4e c2                cmovle eax,edx
  19:   89 45 fc                mov    DWORD PTR [rbp-0x4],eax
```

```
    return a;
  1c:   8b 45 fc                mov    eax,DWORD PTR [rbp-0x4]
  1f:   48 83 c4 10             add    rsp,0x10
  23:   5d                      pop    rbp
  24:   c3                      ret
  25:   90                      nop
  26:   90                      nop
  27:   90                      nop
  28:   90                      nop
  29:   90                      nop
  2a:   90                      nop
  2b:   90                      nop
  2c:   90                      nop
  2d:   90                      nop
  2e:   90                      nop
  2f:   90                      nop

Disassembly of section .xdata:

0000000000000000 <.xdata>:
   0:   01 08                   add    DWORD PTR [rax],ecx
   2:   03 05 08 12 04 03       add    eax,DWORD PTR [rip+0x3041208]        #
3041210 <.xdata+0x3041210>
   8:   01 50 00                add    DWORD PTR [rax+0x0],edx
    ...

Disassembly of section .pdata:

0000000000000000 <.pdata>:
   0:   00 00                   add    BYTE PTR [rax],al
   2:   00 00                   add    BYTE PTR [rax],al
   4:   25 00 00 00 00          and    eax,0x0
   9:   00 00                   add    BYTE PTR [rax],al
    ...

Disassembly of section .debug_frame:

0000000000000000 <.debug_frame>:
   0:   14 00                   adc    al,0x0
{
   2:   00 00                   add    BYTE PTR [rax],al
   4:   ff                      (bad)
   5:   ff                      (bad)
   6:   ff                      (bad)
   7:   ff 03                   inc    DWORD PTR [rbx]
   9:   00 01                   add    BYTE PTR [rcx],al
   b:   78 10                   js     1d <.debug_frame+0x1d>
   d:   0c 07                   or     al,0x7
    a = min(b, c);
   f:   08 90 01 00 00 00       or     BYTE PTR [rax+0x1],dl
  15:   00 00                   add    BYTE PTR [rax],al
  17:   00 34 00                add    BYTE PTR [rax+rax*1],dh
    ...
  26:   00 00                   add    BYTE PTR [rax],al
```

```
  28:   25 00 00 00 00          and    eax,0x0
  2d:   00 00                   add    BYTE PTR [rax],al
  2f:   00 04 01                add    BYTE PTR [rcx+rax*1],al
  32:   00 00                   add    BYTE PTR [rax],al
  34:   00 0e                   add    BYTE PTR [rsi],cl
  36:   10 86 02 04 03 00       adc    BYTE PTR [rsi+0x30402],al
  3c:   00 00                   add    BYTE PTR [rax],al
  3e:   0d 06 04 20 00          or     eax,0x200406
  43:   00 00                   add    BYTE PTR [rax],al
  45:   c6                      (bad)
  46:   0c 07                   or     al,0x7
  48:   08 00                   or     BYTE PTR [rax],al
  4a:   00 00                   add    BYTE PTR [rax],al
  4c:   00 00                   add    BYTE PTR [rax],al
    ...

Disassembly of section .debug_info:

0000000000000000 <.debug_info>:
   0:   a7                      cmps   DWORD PTR ds:[rsi],DWORD PTR es:[rdi]
{
   1:   00 00                   add    BYTE PTR [rax],al
   3:   00 05 00 01 08 00       add    BYTE PTR [rip+0x80100],al        # 80109
<.debug_info+0x80109>
   9:   00 00                   add    BYTE PTR [rax],al
   b:   00 02                   add    BYTE PTR [rdx],al
   d:   47                      rex.RXB
    a = min(b, c);
   e:   4e 55                   rex.WRX push rbp
  10:   20 43 31                and    BYTE PTR [rbx+0x31],al
  13:   37                      (bad)
  14:   20 31                   and    BYTE PTR [rcx],dh
  16:   31 2e                   xor    DWORD PTR [rsi],ebp
  18:   33 2e                   xor    ebp,DWORD PTR [rsi]
  1a:   30 20                   xor    BYTE PTR [rax],ah
    return a;
  1c:   2d 6d 74 75 6e          sub    eax,0x6e75746d
  21:   65 3d 67 65 6e 65       gs cmp eax,0x656e6567
  27:   72 69                   jb     92 <.debug_info+0x92>
  29:   63 20                   movsxd esp,DWORD PTR [rax]
  2b:   2d 6d 61 72 63          sub    eax,0x6372616d
  30:   68 3d 78 38 36          push   0x3638783d
  35:   2d 36 34 20 2d          sub    eax,0x2d203436
  3a:   67 00 1d 1a 00 00 00    add    BYTE PTR [eip+0x1a],bl        # 5b
<.debug_info+0x5b>
    ...
  4d:   25 00 00 00 00          and    eax,0x0
  52:   00 00                   add    BYTE PTR [rax],al
  54:   00 00                   add    BYTE PTR [rax],al
  56:   00 00                   add    BYTE PTR [rax],al
  58:   00 03                   add    BYTE PTR [rbx],al
  5a:   63 6f 6d                movsxd ebp,DWORD PTR [rdi+0x6d]
  5d:   70 61                   jo     c0 <.debug_info+0xc0>
  5f:   72 65                   jb     c6 <.debug_info+0xc6>
```

```
  61:   00 01                   add    BYTE PTR [rcx],al
  63:   02 05 a3 00 00 00       add    al,BYTE PTR [rip+0xa3]        # 10c
<.debug_info+0x10c>
    ...
  71:   25 00 00 00 00          and    eax,0x0
  76:   00 00                   add    BYTE PTR [rax],al
  78:   00 01                   add    BYTE PTR [rcx],al
  7a:   9c                      pushf
  7b:   a3 00 00 00 01 62 00    movabs ds:0xa311006201000000,eax
  82:   11 a3
  84:   00 00                   add    BYTE PTR [rax],al
  86:   00 02                   add    BYTE PTR [rdx],al
  88:   91                      xchg   ecx,eax
  89:   00 01                   add    BYTE PTR [rcx],al
  8b:   63 00                   movsxd eax,DWORD PTR [rax]
  8d:   18 a3 00 00 00 02       sbb    BYTE PTR [rbx+0x2000000],ah
  93:   91                      xchg   ecx,eax
  94:   08 04 61                or     BYTE PTR [rcx+riz*2],al
  97:   00 01                   add    BYTE PTR [rcx],al
  99:   04 09                   add    al,0x9
  9b:   a3 00 00 00 02 91 6c    movabs ds:0x5006c9102000000,eax
  a2:   00 05
  a4:   04 05                   add    al,0x5
  a6:   69                      .byte 0x69
  a7:   6e                      outs   dx,BYTE PTR ds:[rsi]
  a8:   74 00                   je     aa <.debug_info+0xaa>
    ...

Disassembly of section .debug_abbrev:

0000000000000000 <.debug_abbrev>:
   0:   01 05 00 03 08 3a       add    DWORD PTR [rip+0x3a080300],eax        #
3a080306 <.debug_abbrev+0x3a080306>
{
   6:   21 01                   and    DWORD PTR [rcx],eax
   8:   3b 21                   cmp    esp,DWORD PTR [rcx]
   a:   02 39                   add    bh,BYTE PTR [rcx]
   c:   0b 49 13                or     ecx,DWORD PTR [rcx+0x13]
    a = min(b, c);
   f:   02 18                   add    bl,BYTE PTR [rax]
  11:   00 00                   add    BYTE PTR [rax],al
  13:   02 11                   add    dl,BYTE PTR [rcx]
  15:   01 25 08 13 0b 03       add    DWORD PTR [rip+0x30b1308],esp        #
30b1323 <.debug_abbrev+0x30b1323>
  1b:   1f                      (bad)
    return a;
  1c:   1b 1f                   sbb    ebx,DWORD PTR [rdi]
  1e:   11 01                   adc    DWORD PTR [rcx],eax
  20:   12 07                   adc    al,BYTE PTR [rdi]
  22:   10 17                   adc    BYTE PTR [rdi],dl
  24:   00 00                   add    BYTE PTR [rax],al
  26:   03 2e                   add    ebp,DWORD PTR [rsi]
  28:   01 3f                   add    DWORD PTR [rdi],edi
  2a:   19 03                   sbb    DWORD PTR [rbx],eax
```

```
  2c:   08 3a                   or     BYTE PTR [rdx],bh
  2e:   0b 3b                   or     edi,DWORD PTR [rbx]
  30:   0b 39                   or     edi,DWORD PTR [rcx]
  32:   0b 27                   or     esp,DWORD PTR [rdi]
  34:   19 49 13                sbb    DWORD PTR [rcx+0x13],ecx
  37:   11 01                   adc    DWORD PTR [rcx],eax
  39:   12 07                   adc    al,BYTE PTR [rdi]
  3b:   40 18 7a 19             sbb    BYTE PTR [rdx+0x19],dil
  3f:   01 13                   add    DWORD PTR [rbx],edx
  41:   00 00                   add    BYTE PTR [rax],al
  43:   04 34                   add    al,0x34
  45:   00 03                   add    BYTE PTR [rbx],al
  47:   08 3a                   or     BYTE PTR [rdx],bh
  49:   0b 3b                   or     edi,DWORD PTR [rbx]
  4b:   0b 39                   or     edi,DWORD PTR [rcx]
  4d:   0b 49 13                or     ecx,DWORD PTR [rcx+0x13]
  50:   02 18                   add    bl,BYTE PTR [rax]
  52:   00 00                   add    BYTE PTR [rax],al
  54:   05 24 00 0b 0b          add    eax,0xb0b0024
  59:   3e 0b 03                ds or  eax,DWORD PTR [rbx]
  5c:   08 00                   or     BYTE PTR [rax],al
    ...

Disassembly of section .debug_aranges:

0000000000000000 <.debug_aranges>:
   0:   2c 00                   sub    al,0x0
{
   2:   00 00                   add    BYTE PTR [rax],al
   4:   02 00                   add    al,BYTE PTR [rax]
   6:   00 00                   add    BYTE PTR [rax],al
   8:   00 00                   add    BYTE PTR [rax],al
   a:   08 00                   or     BYTE PTR [rax],al
    ...
    a = min(b, c);
  18:   25 00 00 00 00          and    eax,0x0
    ...

Disassembly of section .debug_line:

0000000000000000 <.debug_line>:
   0:   4e 00 00                rex.WRX add BYTE PTR [rax],r8b
{
   3:   00 05 00 08 00 2a       add    BYTE PTR [rip+0x2a000800],al        #
2a000809 <.debug_line+0x2a000809>
   9:   00 00                   add    BYTE PTR [rax],al
   b:   00 01                   add    BYTE PTR [rcx],al
   d:   01 01                   add    DWORD PTR [rcx],eax
    a = min(b, c);
   f:   fb                      sti
  10:   0e                      (bad)
  11:   0d 00 01 01 01          or     eax,0x1010100
  16:   01 00                   add    DWORD PTR [rax],eax
  18:   00 00                   add    BYTE PTR [rax],al
```

```
  1a:   01 00                   add    DWORD PTR [rax],eax
    return a;
  1c:   00 01                   add    BYTE PTR [rcx],al
  1e:   01 01                   add    DWORD PTR [rcx],eax
  20:   1f                      (bad)
  21:   01 22                   add    DWORD PTR [rdx],esp
  23:   00 00                   add    BYTE PTR [rax],al
  25:   00 02                   add    BYTE PTR [rdx],al
  27:   01 1f                   add    DWORD PTR [rdi],ebx
  29:   02 0f                   add    cl,BYTE PTR [rdi]
  2b:   02 3c 00                add    bh,BYTE PTR [rax+rax*1]
  2e:   00 00                   add    BYTE PTR [rax],al
  30:   00 44 00 00             add    BYTE PTR [rax+rax*1+0x0],al
  34:   00 00                   add    BYTE PTR [rax],al
  36:   05 01 00 09 02          add    eax,0x2090001
    ...
  43:   14 05                   adc    al,0x5
  45:   07                      (bad)
  46:   d8 05 0c d7 05 01       fadd   DWORD PTR [rip+0x105d70c]        # 105d758
<.debug_line+0x105d758>
  4c:   3d 02 06 00 01          cmp    eax,0x1000602
  51:   01                      .byte 0x1

Disassembly of section .debug_line_str:

0000000000000000 <.debug_line_str>:
   0:   2f                      (bad)
{
   1:   68 6f 6d 65 2f          push   0x2f656d6f
   6:   6a 69                   push   0x69
   8:   77 6f                   ja     79 <.debug_line_str+0x79>
   a:   6e                      outs   dx,BYTE PTR ds:[rsi]
   b:   2f                      (bad)
   c:   77 6f                   ja     7d <.debug_line_str+0x7d>
    a = min(b, c);
   e:   72 6b                   jb     7b <.debug_line_str+0x7b>
  10:   73 70                   jae    82 <.debug_line_str+0x82>
  12:   61                      (bad)
  13:   63 65 2f                movsxd esp,DWORD PTR [rbp+0x2f]
  16:   78 38                   js     50 <.debug_line_str+0x50>
  18:   36 00 63 6f             ss add BYTE PTR [rbx+0x6f],ah
    return a;
  1c:   64 65 31 2e             fs xor DWORD PTR gs:[rsi],ebp
  20:   63 00                   movsxd eax,DWORD PTR [rax]
  22:   2f                      (bad)
  23:   68 6f 6d 65 2f          push   0x2f656d6f
  28:   6a 69                   push   0x69
  2a:   77 6f                   ja     9b <.debug_line_str+0x9b>
  2c:   6e                      outs   dx,BYTE PTR ds:[rsi]
  2d:   2f                      (bad)
  2e:   77 6f                   ja     9f <.debug_line_str+0x9f>
  30:   72 6b                   jb     9d <.debug_line_str+0x9d>
  32:   73 70                   jae    a4 <.debug_line_str+0xa4>
  34:   61                      (bad)
```

```
  35:   63 65 2f                movsxd esp,DWORD PTR [rbp+0x2f]
  38:   78 38                   js     72 <.debug_line_str+0x72>
  3a:   36 00 63 6f             ss add BYTE PTR [rbx+0x6f],ah
  3e:   64 65 31 2e             fs xor DWORD PTR gs:[rsi],ebp
  42:   63 00                   movsxd eax,DWORD PTR [rax]
  44:   63 6f 64                movsxd ebp,DWORD PTR [rdi+0x64]
  47:   65 31 2e                xor    DWORD PTR gs:[rsi],ebp
  4a:   63 00                   movsxd eax,DWORD PTR [rax]

Disassembly of section .rdata$zzz:

0000000000000000 <.rdata$zzz>:
   0:   47                      rex.RXB
   1:   43                      rex.XB
   2:   43 3a 20                rex.XB cmp spl,BYTE PTR [r8]
   5:   28 47 4e                sub    BYTE PTR [rdi+0x4e],al
   8:   55                      push   rbp
   9:   29 20                   sub    DWORD PTR [rax],esp
   b:   31 31                   xor    DWORD PTR [rcx],esi
   d:   2e 33 2e                cs xor ebp,DWORD PTR [rsi]
  10:   30 00                   xor    BYTE PTR [rax],al
    ...
```

3. Outputs of each compilation step in the RISC-V cross-compilation

Preprocessor : code1.i

```
# 1 "code1.c"
# 1 "<built-in>"
# 1 "<command-line>"
# 1 "code1.c"

int compare(int b, int c)
{
    int a;
    a = ((b) < (c) ? (b) : (c));;
    return a;
}
```

Compilation : code1.s

```
    .file   "code1.c"
    .option nopic
    .attribute arch, "rv32i2p0"
```

```
    .attribute unaligned_access, 0
    .attribute stack_align, 16
    .text
.Ltext0:
    .cfi_sections   .debug_frame
    .align  2
    .globl  compare
    .type   compare, @function
compare:
.LFB0:
    .file 1 "code1.c"
    .loc 1 3 1
    .cfi_startproc
    addi    sp,sp,-48
    .cfi_def_cfa_offset 48
    sw  s0,44(sp)
    .cfi_offset 8, -4
    addi    s0,sp,48
    .cfi_def_cfa 8, 0
    sw  a0,-36(s0)
    sw  a1,-40(s0)
    .loc 1 5 7
    lw  a4,-36(s0)
    lw  a5,-40(s0)
    ble a5,a4,.L2
    mv  a5,a4
.L2:
    sw  a5,-20(s0)
    .loc 1 6 12
    lw  a5,-20(s0)
    .loc 1 7 1
    mv  a0,a5
    lw  s0,44(sp)
    .cfi_restore 8
    .cfi_def_cfa 2, 48
    addi    sp,sp,48
    .cfi_def_cfa_offset 0
    jr  ra
    .cfi_endproc
.LFE0:
    .size   compare, .-compare
.Letext0:
    .section    .debug_info,"",@progbits
.Ldebug_info0:
    .4byte  0x6b
    .2byte  0x4
    .4byte  .Ldebug_abbrev0
    .byte   0x4
    .byte   0x1
    .4byte  .LASF0
    .byte   0xc
    .4byte  .LASF1
    .4byte  .LASF2
    .4byte  .Ltext0
```

```
    .4byte  .Letext0-.Ltext0
    .4byte  .Ldebug_line0
    .byte   0x2
    .4byte  .LASF3
    .byte   0x1
    .byte   0x2
    .byte   0x5
    .4byte  0x67
    .4byte  .LFB0
    .4byte  .LFE0-.LFB0
    .byte   0x1
    .byte   0x9c
    .4byte  0x67
    .byte   0x3
    .string "b"
    .byte   0x1
    .byte   0x2
    .byte   0x11
    .4byte  0x67
    .byte   0x2
    .byte   0x91
    .byte   0x5c
    .byte   0x3
    .string "c"
    .byte   0x1
    .byte   0x2
    .byte   0x18
    .4byte  0x67
    .byte   0x2
    .byte   0x91
    .byte   0x58
    .byte   0x4
    .string "a"
    .byte   0x1
    .byte   0x4
    .byte   0x9
    .4byte  0x67
    .byte   0x2
    .byte   0x91
    .byte   0x6c
    .byte   0
    .byte   0x5
    .byte   0x4
    .byte   0x5
    .string "int"
    .byte   0
    .section    .debug_abbrev,"",@progbits
.Ldebug_abbrev0:
    .byte   0x1
    .byte   0x11
    .byte   0x1
    .byte   0x25
    .byte   0xe
    .byte   0x13
```

```
	.byte	0xb
	.byte	0x3
	.byte	0xe
	.byte	0x1b
	.byte	0xe
	.byte	0x11
	.byte	0x1
	.byte	0x12
	.byte	0x6
	.byte	0x10
	.byte	0x17
	.byte	0
	.byte	0
	.byte	0x2
	.byte	0x2e
	.byte	0x1
	.byte	0x3f
	.byte	0x19
	.byte	0x3
	.byte	0xe
	.byte	0x3a
	.byte	0xb
	.byte	0x3b
	.byte	0xb
	.byte	0x39
	.byte	0xb
	.byte	0x27
	.byte	0x19
	.byte	0x49
	.byte	0x13
	.byte	0x11
	.byte	0x1
	.byte	0x12
	.byte	0x6
	.byte	0x40
	.byte	0x18
	.byte	0x97,0x42
	.byte	0x19
	.byte	0x1
	.byte	0x13
	.byte	0
	.byte	0
	.byte	0x3
	.byte	0x5
	.byte	0
	.byte	0x3
	.byte	0x8
	.byte	0x3a
	.byte	0xb
	.byte	0x3b
	.byte	0xb
	.byte	0x39
	.byte	0xb
	.byte	0x49
```

```
    .byte   0x13
    .byte   0x2
    .byte   0x18
    .byte   0
    .byte   0
    .byte   0x4
    .byte   0x34
    .byte   0
    .byte   0x3
    .byte   0x8
    .byte   0x3a
    .byte   0xb
    .byte   0x3b
    .byte   0xb
    .byte   0x39
    .byte   0xb
    .byte   0x49
    .byte   0x13
    .byte   0x2
    .byte   0x18
    .byte   0
    .byte   0
    .byte   0x5
    .byte   0x24
    .byte   0
    .byte   0xb
    .byte   0xb
    .byte   0x3e
    .byte   0xb
    .byte   0x3
    .byte   0x8
    .byte   0
    .byte   0
    .byte   0
    .section    .debug_aranges,"",@progbits
    .4byte  0x1c
    .2byte  0x2
    .4byte  .Ldebug_info0
    .byte   0x4
    .byte   0
    .2byte  0
    .2byte  0
    .4byte  .Ltext0
    .4byte  .Letext0-.Ltext0
    .4byte  0
    .4byte  0
    .section    .debug_line,"",@progbits
.Ldebug_line0:
    .section    .debug_str,"MS",@progbits,1
.LASF2:
    .string "/home/jiwon/workspace"
.LASF0:
    .string "GNU C17 10.1.0 -march=rv32i -mtune=rocket -mabi=ilp32 -g"
.LASF1:
```

```
    .string "code1.c"
.LASF3:
    .string "compare"
    .ident  "GCC: (GNU) 10.1.0"
```

Assembler : code1.dump

```
code1.o:     file format elf32-littleriscv
code1.o
architecture: riscv:rv32, flags 0x00000011:
HAS_RELOC, HAS_SYMS
start address 0x00000000

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000003c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  00000070  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  00000070  2**0
                  ALLOC
  3 .debug_info   0000006f  00000000  00000000  00000070  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  4 .debug_abbrev 0000005f  00000000  00000000  000000df  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_aranges 00000020  00000000  00000000  0000013e  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  6 .debug_line   00000050  00000000  00000000  0000015e  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  7 .debug_str    0000005f  00000000  00000000  000001ae  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  8 .comment      00000013  00000000  00000000  0000020d  2**0
                  CONTENTS, READONLY
  9 .debug_frame  00000034  00000000  00000000  00000220  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
 10 .riscv.attributes 0000001c  00000000  00000000  00000254  2**0
                  CONTENTS, READONLY
SYMBOL TABLE:
00000000 l    df *ABS*  00000000 code1.c
00000000 l    d  .text  00000000 .text
00000000 l    d  .data  00000000 .data
00000000 l    d  .bss   00000000 .bss
00000000 l       .text  00000000 .L0
00000000 l       .text  00000000 .L0
0000000c l       .text  00000000 .L0
00000014 l       .text  00000000 .L0
00000028 l       .text  00000000 .L0
0000002c l       .text  00000000 .L0
00000034 l       .text  00000000 .L0
```

```
0000003c l       .text  00000000 .L0
00000000 l    d  .debug_info    00000000 .debug_info
00000000 l    d  .debug_abbrev  00000000 .debug_abbrev
00000000 l    d  .debug_aranges 00000000 .debug_aranges
00000000 l    d  .debug_line    00000000 .debug_line
00000000 l    d  .debug_str 00000000 .debug_str
0000003c l       .text  00000000 .L0
00000000 l       .debug_frame   00000000 .L0
00000024 l       .text  00000000 .L2
00000000 l       .debug_abbrev  00000000 .Ldebug_abbrev0
00000016 l       .debug_str 00000000 .LASF0
0000004f l       .debug_str 00000000 .LASF1
00000000 l       .debug_str 00000000 .LASF2
00000000 l       .text  00000000 .Ltext0
0000003c l       .text  00000000 .Letext0
00000000 l       .debug_line    00000000 .Ldebug_line0
00000057 l       .debug_str 00000000 .LASF3
00000000 l       .text  00000000 .LFB0
0000003c l       .text  00000000 .LFE0
00000000 l       .debug_info    00000000 .Ldebug_info0
00000000 l    d  .comment   00000000 .comment
00000000 l    d  .debug_frame   00000000 .debug_frame
00000000 l    d  .riscv.attributes  00000000 .riscv.attributes
00000000 g     F .text  0000003c compare



Disassembly of section .text:

00000000 <compare>:
#define min(x, y) ((x) < (y) ? (x) : (y));
int compare(int b, int c)
{
   0:   fd010113            addi    sp,sp,-48
   4:   02812623            sw  s0,44(sp)
   8:   03010413            addi    s0,sp,48
   c:   fca42e23            sw  a0,-36(s0)
  10:   fcb42c23            sw  a1,-40(s0)
    int a;
    a = min(b, c);
  14:   fdc42703            lw  a4,-36(s0)
  18:   fd842783            lw  a5,-40(s0)
  1c:   00f75463            bge a4,a5,24 <.L2>
            1c: R_RISCV_BRANCH  .L2
  20:   00070793            mv  a5,a4

00000024 <.L2>:
  24:   fef42623            sw  a5,-20(s0)
    return a;
  28:   fec42783            lw  a5,-20(s0)
  2c:   00078513            mv  a0,a5
  30:   02c12403            lw  s0,44(sp)
  34:   03010113            addi    sp,sp,48
```

```
  38:   00008067            ret
```

Linker : labcode.dump

```
labcode:     file format elf32-littleriscv
labcode
architecture: riscv:rv32, flags 0x00000012:
EXEC_P, HAS_SYMS
start address 0x00000000

Program Header:
    LOAD off    0x00000060 vaddr 0x00000000 paddr 0x00000000 align 2**4
         filesz 0x00000800 memsz 0x00000800 flags rwx

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000004c  00000000  00000000  00000060  2**4
                  CONTENTS, ALLOC, LOAD, CODE
  1 .data         00000400  00000400  00000400  00000460  2**4
                  CONTENTS, ALLOC, LOAD, DATA
  2 .riscv.attributes 0000001c  00000000  00000000  00000860  2**0
                  CONTENTS, READONLY
  3 .comment      00000012  00000000  00000000  0000087c  2**0
                  CONTENTS, READONLY
  4 .debug_line   0000008b  00000000  00000000  0000088e  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_info   00000095  00000000  00000000  00000919  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_abbrev 00000073  00000000  00000000  000009ae  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  7 .debug_aranges 00000040  00000000  00000000  00000a28  2**3
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  8 .debug_str    00000072  00000000  00000000  00000a68  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  9 .debug_frame  00000034  00000000  00000000  00000adc  2**2
                  CONTENTS, READONLY, DEBUGGING, OCTETS
SYMBOL TABLE:
00000000 l    d  .text  00000000 .text
00000400 l    d  .data  00000000 .data
00000000 l    d  .riscv.attributes  00000000 .riscv.attributes
00000000 l    d  .comment   00000000 .comment
00000000 l    d  .debug_line    00000000 .debug_line
00000000 l    d  .debug_info    00000000 .debug_info
00000000 l    d  .debug_abbrev  00000000 .debug_abbrev
00000000 l    d  .debug_aranges 00000000 .debug_aranges
00000000 l    d  .debug_str 00000000 .debug_str
00000000 l    d  .debug_frame   00000000 .debug_frame
00000000 l    df *ABS*  00000000 lab0.o
00000400 l       .data  00000000 stack
```

```
00000000 l    df *ABS*  00000000 code1.c
00000010 g     F .text  0000003c compare



Disassembly of section .text:

00000000 <compare-0x10>:
.text
.align 4

    la sp, stack
   0:   40000113            li  sp,1024
    j   compare
   4:   00c0006f            j   10 <compare>
    ...

00000010 <compare>:
#define min(x, y) ((x) < (y) ? (x) : (y));
int compare(int b, int c)
{
  10:   fd010113            addi    sp,sp,-48
  14:   02812623            sw  s0,44(sp)
  18:   03010413            addi    s0,sp,48
  1c:   fca42e23            sw  a0,-36(s0)
  20:   fcb42c23            sw  a1,-40(s0)
    int a;
    a = min(b, c);
  24:   fdc42703            lw  a4,-36(s0)
  28:   fd842783            lw  a5,-40(s0)
  2c:   00f75463            bge a4,a5,34 <compare+0x24>
  30:   00070793            mv  a5,a4
  34:   fef42623            sw  a5,-20(s0)
    return a;
  38:   fec42783            lw  a5,-20(s0)
  3c:   00078513            mv  a0,a5
  40:   02c12403            lw  s0,44(sp)
  44:   03010113            addi    sp,sp,48
  48:   00008067            ret
```