Hitachi America, Ltd.

TN-0156

Application Engineering
TechNote

Jennifer Ediyanto

Bit Clear Using C or Inline Assembly

The GNU C Compiler allows us to clear a bit by using the pointer in C code or inline Assembly. This paper will provide sample codes in clearing bits.

The following are bit declarations:

```
struct
{
   int  smr:1;
   int  scr:1;
   int  ssr:1;
   int  tdr:1;
} *sci0;
```

The above is a pointer-type structure declaration that consists of 4 integer-type elements. The pointer structure is named as sci0 and the elements are named as smr, scr, ssr, and tdr. Each element is one bit wide with the order from the 7th bit to 4th bit, i.e., smr is the 7th bit. The rest of the bits (3rd bit to 0th bit) are undefined in this structure.

```
struct
{
   char a;
   char b;
} *j;
```

The above is a pointer-type structure declaration that consists of 2 character-type elements. The pointer structure is named as 'j' and the elements are named as 'a' and 'b'. Each element is one bit wide with order from the 7th bit to 6th bit, i.e., 'a' is the 7th bit. The rest of the bits (5th bit to 0th bit) are undefined in this structure.

The following are three ways to clear the bit:

```
b()
{
  asm("bclr #7,%X0" : "=U,r" (j->a));
}
```

The function 'b' contains an inline assembly code to clear the 7th bit of 'j' structure, which is element 'a'. Some symbols in the inline assembly mean:

bclr	bit clear instruction
#7	the 7th bit
%X0	print as byte register
=U	operand memory reference
r	use as a register
j->a	element 'a' of 'j' structure

```
m()
{
  sci0->ssr = 0;
}
```

The function 'm' contains a C statement to clear ssr bit in the sci0 structure. By assigning zero to the sci0 pointer to ssr, this statement will clear ssr bit (the 5th bit).

```
a()
{
  j->b &= ~0x40;
}
```

The function 'a' contains a C statement to clear 'b' bit in the 'j' structure. By using the logical-and (&) with the value of 'not 0x40 (i.e., 0x7F), this statement will clear 'b' bit (the 6th bit) of 'j' structure.

The following are the generated Assembly source:

```
  16:t.c           **** b()
  17:t.c           **** {
  18:t.c           ****   asm("bclr #7,%X0" : "=U,r" (j->a));
  79 0000 6B020000             mov.w   @_j,r2
  80                       ; #APP
  81 0004 7D207260             bclr    #7,@r2
  82                       ; #NO_APP
  19:t.c           **** }
  89 0008 5470                 rts

  21:t.c           **** m()
  22:t.c           **** {
  23:t.c           ****   sci0->ssr = 0;
 108 000c 6B020000             mov.w   @_sci0,r2
 109 0010 7D207250             bclr    #5,@r2
  24:t.c           **** }
 116 0014 5470                 rts

  27:t.c           **** a()
  28:t.c           **** {
  29:t.c           ****   j->b &= ~0x40;
 135 0018 6B020000             mov.w   @_j,r2
 136 001c 0B02                 adds    #1,r2
 137 001e 7D207260             bclr    #6,@r2
  30:t.c           **** }
 144 0022 5470                 rts
```


HITACHI
Hitachi America, Ltd. • San Francisco Center • 2000 Sierra Point Parkway • Brisbane, CA 94005-1819 • (415) 589-8300