
EXAMPLE 7

Description:
Creating own "prolog" and "epilog" to a function without
implicit entry and return parameter handling.

Notes:

GCC example solves thumbs-interwork by means of proprietary inline assembler code and naked function.

IAR - EWARM has a checkbox called Interwork (option --interwork) which makes it possible to mix files compiled as arm and thumb as long as they are all compiled with the --interwork option.

Danger warning for IAR option: LOW

GCC code example:

```
#define ISR_ENTRY() asm volatile(" sub   lr, lr,#4\n" \
                                 " stmfd sp!,{r0-r12,lr}\n" \
                                 " mrs   r1, spsr\n" \
                                 " stmfd sp!,{r1}")
#define ISR_EXIT()  asm volatile(" ldmfd sp!,{r1}\n" \
                                 " msr   spsr_c,r1\n" \
                                 " ldmfd sp!,{r0-r12,pc}^")

void serial0_isr (void)   __attribute__ ((naked));

void serial0_isr(void)
{
  static unsigned short Var16;
  // perform proper ISR entry so thumb-interwork works properly
  ISR_ENTRY();
  Var16++;
  // recover registers and return
  ISR_EXIT();
}
```

GCC assembler output:

```
           .align     2
           .global    serial0_isr
           .type      serial0_isr,function
serial0_isr:
           @ Naked Function: prologue and epilogue provided by programmer.
           @ args = 0, pretend = 0, frame = 0
           @ frame_needed = 0, current_function_anonymous_args = 0
            sub   lr, lr,#4
 stmfd sp!,{r0-r12,lr}
 mrs   r1, spsr
 stmfd sp!,{r1}
           ldr        r2, .L15
           ldrh       r3, [r2, #0]         @ movhi
           add        r3, r3, #1
           strh       r3, [r2, #0]         @ movhi
            ldmfd sp!,{r1}
 msr   spsr_c,r1
 ldmfd sp!,{r0-r12,pc}^
```

Acknowledgment: Ivan Paulik @ PASAT, Bratislava

```
.L16:
           .align     2
.L15:
           .word      Var16.1
.Lfe6:
           .size      serial0_isr,.Lfe6-serial0_isr
```

Equivalent IAR C code:

```
void serial0_isr(void)
{
  static unsigned short Var16;

  Var16++;
}
```

IAR assembler output:

```
      1          void serial0_isr(void)
      2          {
      3            static unsigned short Var16;
      4
      5            Var16++;
   \                     serial0_isr:
   \   00000000   ....               LDR      R0,??DataTable0
   \   00000002   0088               LDRH     R0,[R0, #+0]
   \   00000004   401C               ADDS     R0,R0,#+1
   \   00000006   ....               LDR      R1,??DataTable0
   \   00000008   0880               STRH     R0,[R1, #+0]
      6          }
   \   0000000A   7047               BX       LR               ;; return

   \                                 In section .bss, align 2
   \                     ??Var16:
   \   00000000                      DS8 2
```

Acknowledgment: Ivan Paulik @ PASAT, Bratislava