

Running Portion of the Application in RAM

The present technical note describes how to run part of an application from RAM. This technical note is valid for tools running linker V5.0.19 or higher (HC08 V2.1.x or higher or HC12 V2.0). This can be used for instance to copy flash programming applet to RAM prior to executing them.

Implementing code to be Copied

Allocate Code to be Executed from RAM in a user define code segment.

This is done in the following way:

```
#pragma CODE_SEG ToCopyToRAM
void func(void) {
    /* Insert code here */
}
#pragma CODE_SEG DEFAULT;
```

If you need to execute more than one function from RAM, you can define all the functions in the same segment. The copy will be performed on a segment base.

Adjust PRM file

The programmer must then specify where the segment must be allocated initially and where it has to be copied.

This is done in the PRM file. Example:

```
NAMES  END

SECTIONS
    Z_RAM      = READ_WRITE  0x0080 TO 0x00FF;
    MY_RAM     = READ_WRITE  0x0100 TO 0x01FF;
    MY_ROM     = READ_ONLY   0xF300 TO 0xFEFF;
    ROM_IMAGE = READ_ONLY   0xF000 TO 0xF2FF RELOCATE_TO 0x0200;

PLACEMENT
    DEFAULT_ROM      INTO  MY_ROM;
    DEFAULT_RAM      INTO  MY_RAM;
    ToCopyToRAM      INTO  ROM_IMAGE;

END

STACKSIZE 0x60
VECTOR 0 _Startup
```

“ToCopyToRAM INTO ROM_IMAGE”

Specifies that the user defined segment ToCopyToRAM should be allocated in ROM_IMAGE section.

“ROM_IMAGE = READ_ONLY 0xF000 TO 0xF2FF RELOCATE_TO 0x0200;”

Specifies that the section ROM_IMAGE must be initially allocated at address 0xF000 and will be relocated to address 0x200.

Implement Copy to RAM function

Customer has to implement the function which will copy the code from ROM to RAM.

We propose the following implementation:

```
extern char __SEG_START_ToCopyToRAM[];
extern char __SEG_SIZE_ToCopyToRAM[];

void func(void); /* Prototype from first function in segment
                  to be copied in RAM. */

/*
   Start_Copy_In_RAM refers to the beginning of the segment
   ToCopyToRAM. This segment contains the functions after
   they have been copied to RAM.
*/
#define Start_Copy_In_RAM __SEG_START_ToCopyToRAM
#define Size_Copy_In_RAM __SEG_SIZE_ToCopyToRAM
/*
   Start_In_ROM refers to the beginning of the segment
   ToCopyToRAM. This segment contains the functions in ROM.
*/

void CopyInRAM(void) {

    char *srcPtr, *dstPtr;
    int count;

    srcPtr = (char *)Start_Copy_In_RAM;
    dstPtr = (char *)&func;

    for (count = 0; count < (int) Size_Copy_In_RAM;
         count++, dstPtr++, srcPtr++) {
        *dstPtr = *srcPtr;
    }
}
```

Just invoke the CopyInRAM function before executing the code from RAM. Any invocation from function func1 will result in a JSR 0x200.