

---

## Execute Code on SRAM

---

**2024.01.05**

### **Introduction**

This document introduces how to execute code on MCU embedded SRAM by an UART communication project example with KEIL MDK toolchain for megawin MG32-Series devices.

User needs to plan the C code files' location or functions' location. And user must assign these files or functions to SRAM or Flash memory space on KEIL IDE.

### **Apply To**

MG32F02A128/U128/A064/U064/A032/V032.

### **Method**

There are two methods. One is directly to put one .C file or some .C files to separated SRAM or Flash memory space. Two is to put separated .C Functions to separated SRAM or Flash memory space.

#### **1. C Source Files**

1. Reference Figure 1, Set Memory configuration, Figure 2.
2. Assign .c file to designated memory space, Figure 3 to 6.

Figure 1. Flash Memory Address Mapping with Normal.

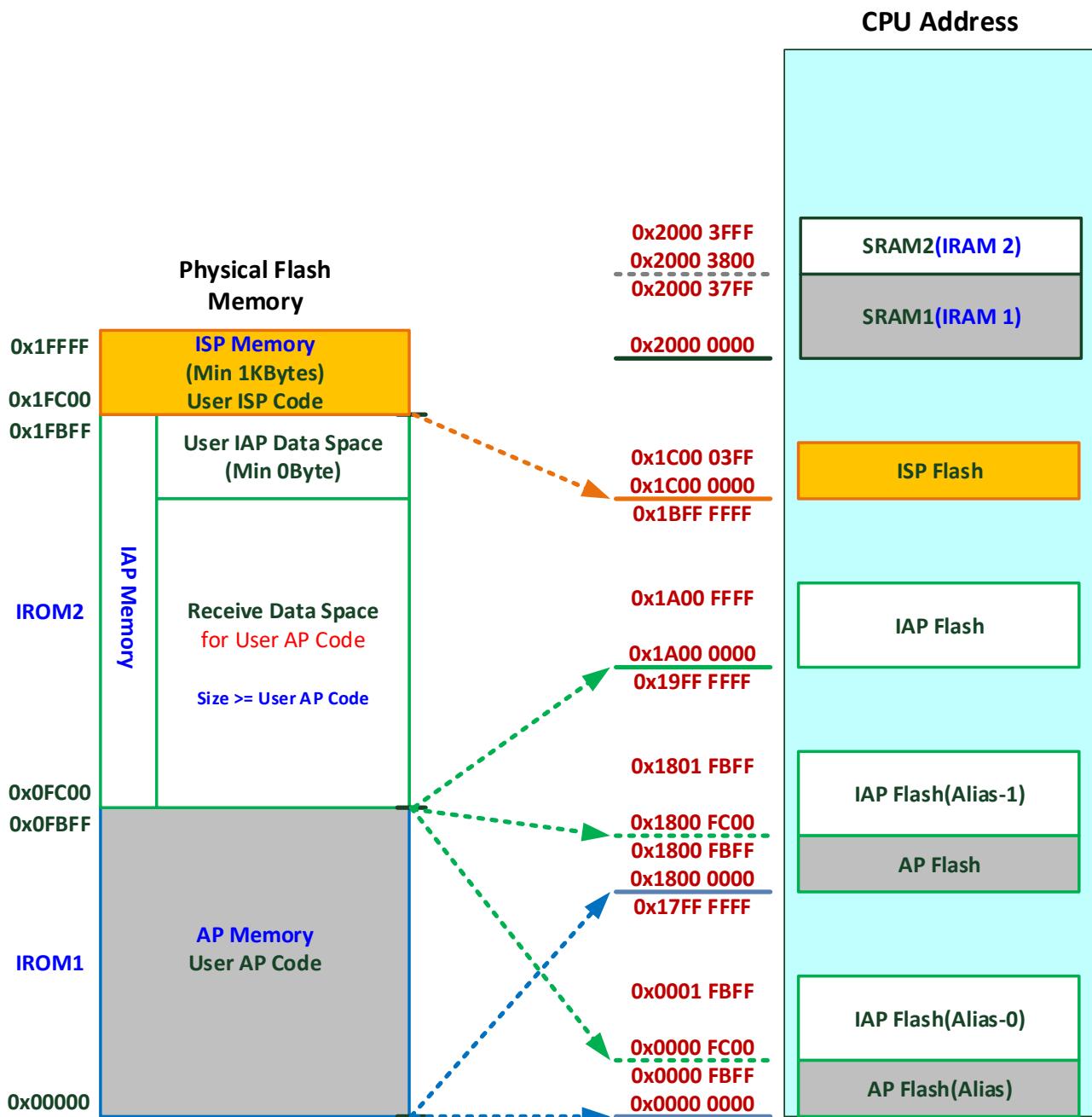


Figure 2. Keil MDK Memory Configuration-1.

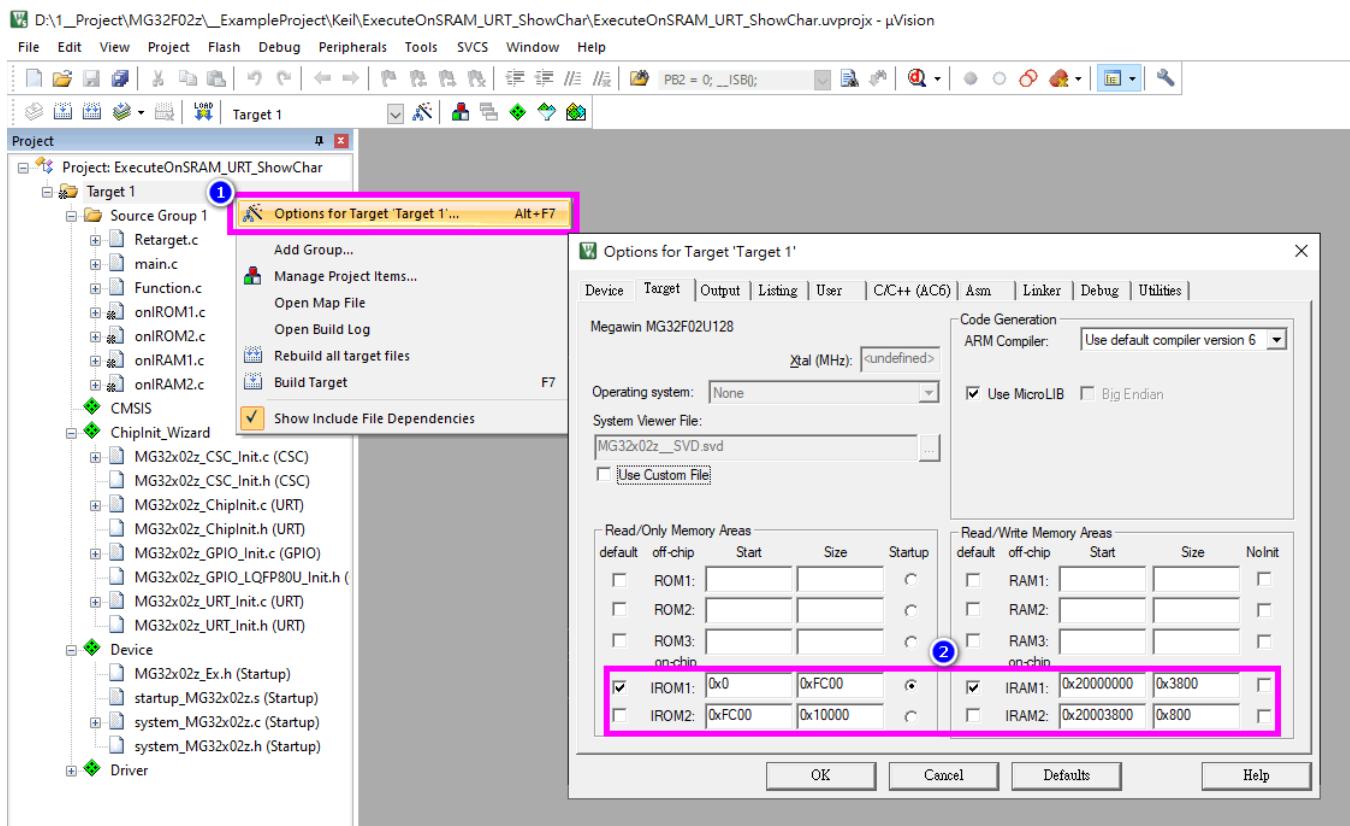


Figure 3. Keil MDK Configuration for onIROM1.c File to IROM1.

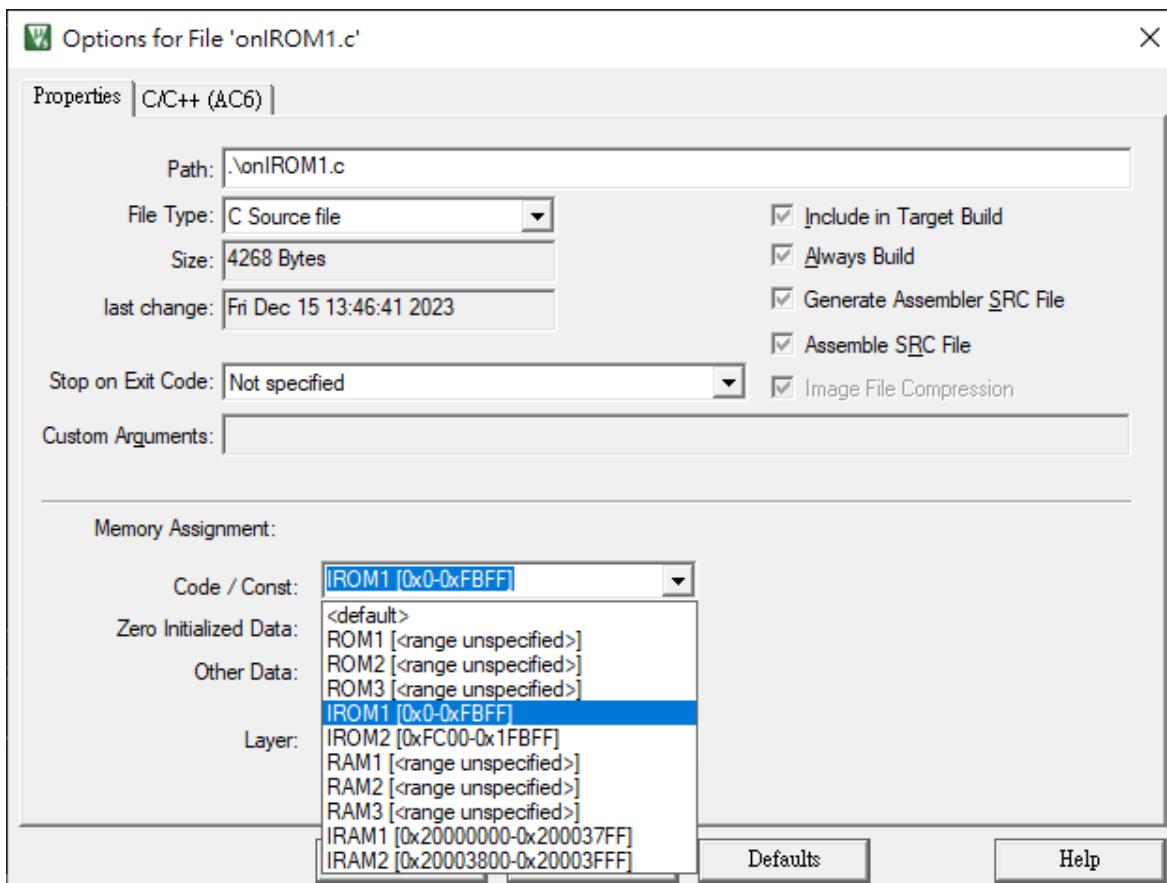


Figure 4. Keil MDK Configuration for onIROM2.c File to IROM2.

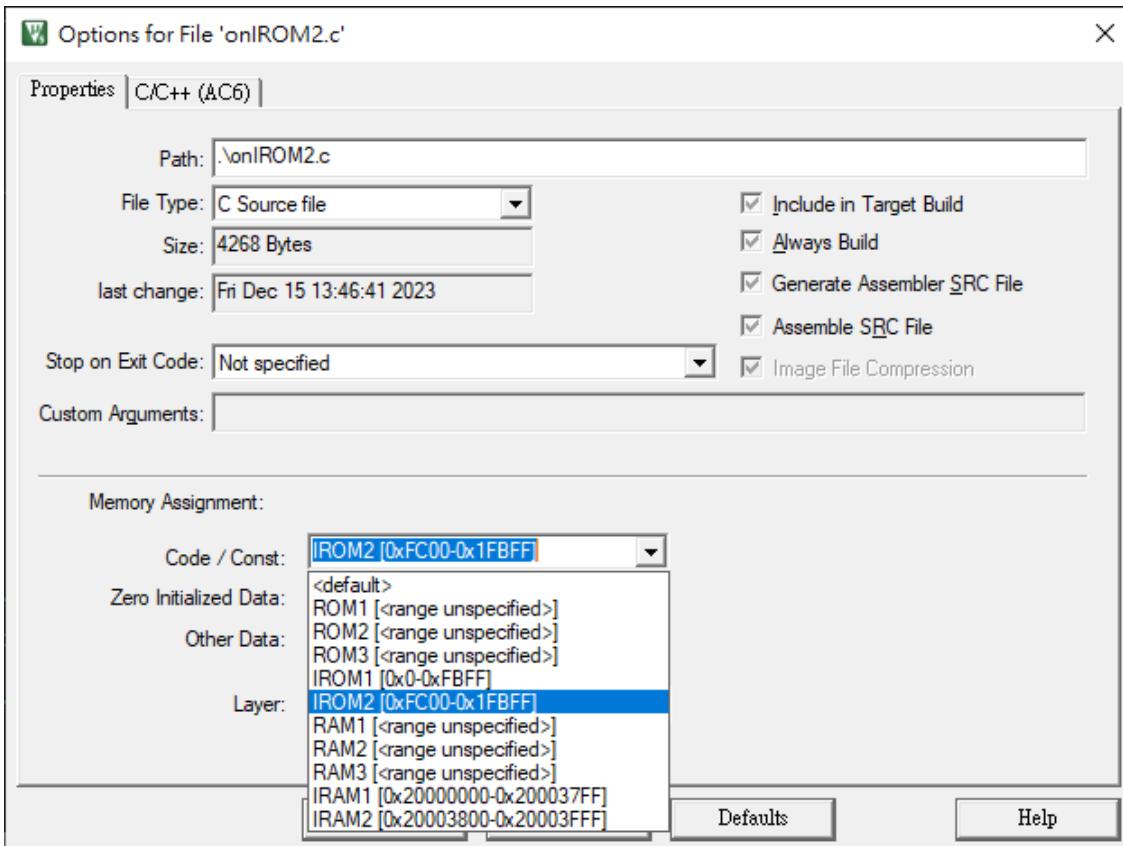


Figure 5. Keil MDK Configuration for onIRAM1.c File to IRAM1.

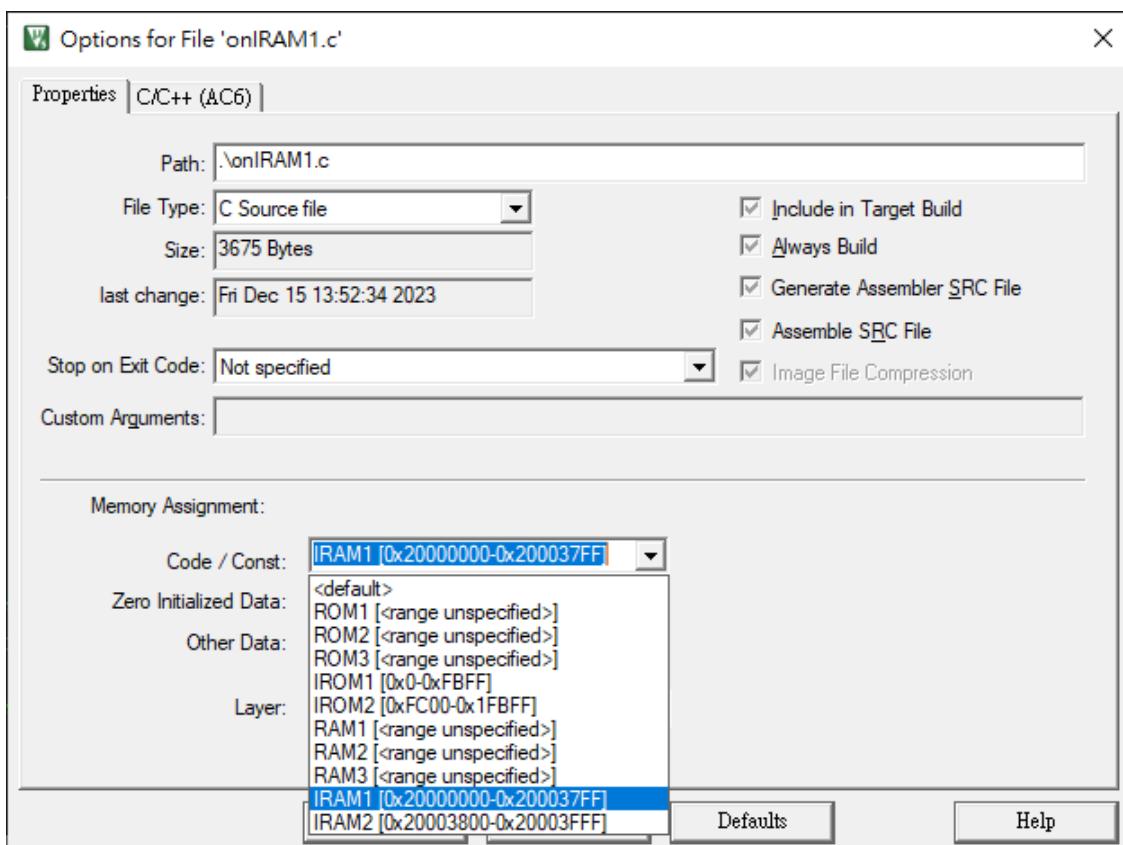
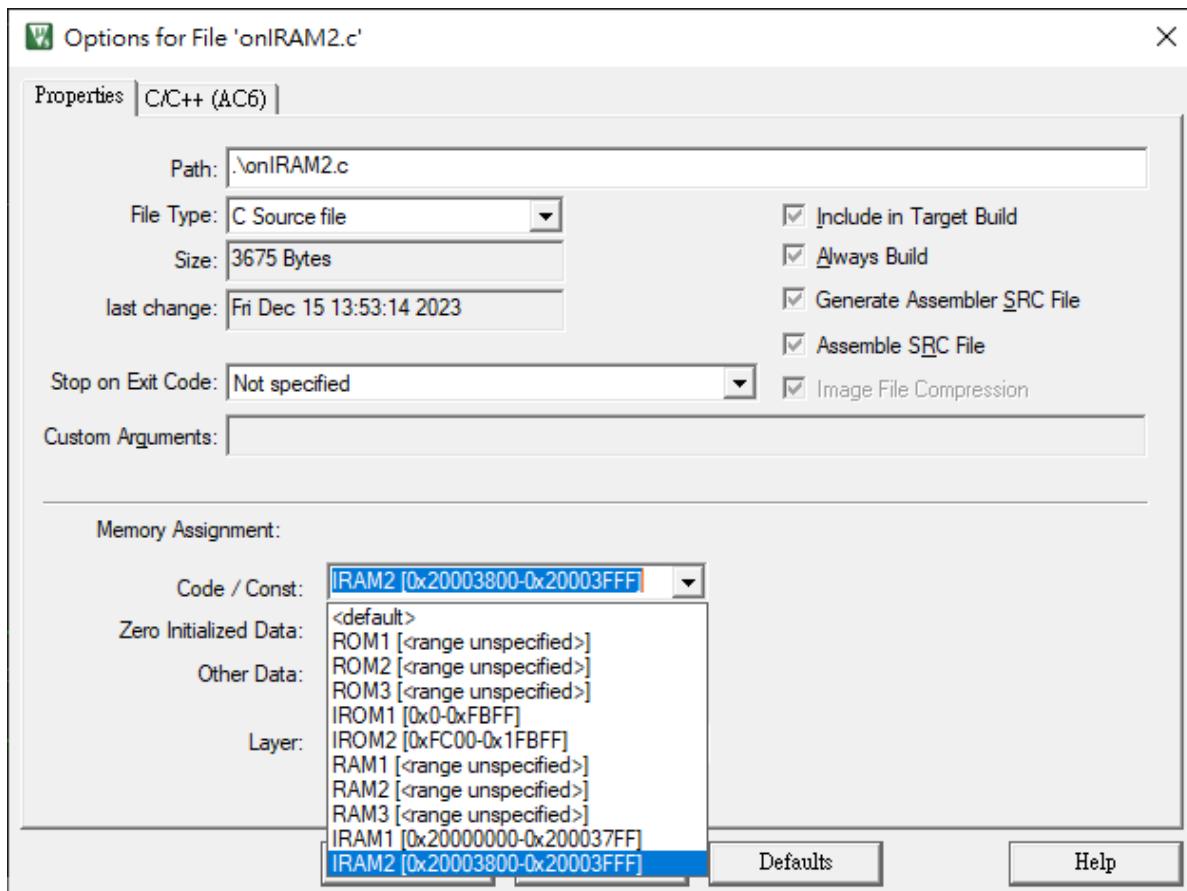


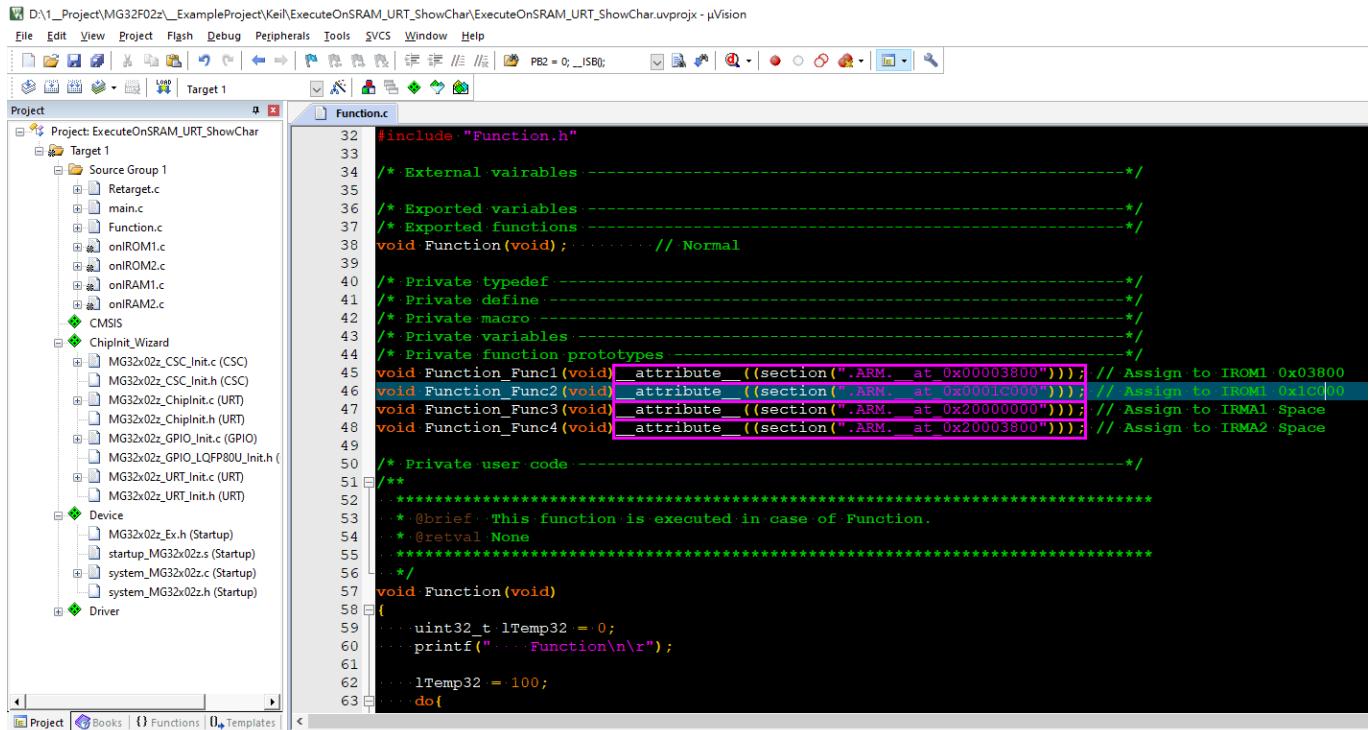
Figure 6. Keil MDK Configuration for onIRAM2.c File to IRAM2.



## 2. C Functions

1. Reference Figure 1, Set Memory configuration, Figure 2.
2. Assign .c file to designated memory space, Figure 7.

**Figure 7. Function assign to designated address.(for Keil V6 Compiler)**



The screenshot shows the Keil V6 IDE interface with the 'Function.c' file open in the editor. The code contains several instances of the `__attribute__((section(".ARM._at_Address")))` keyword to specify memory locations for different functions. The project tree on the left shows various source files and library components for a MG32F02z project.

```

32 #include "Function.h"
33
34 /* External variables */
35
36 /* Exported variables */
37 /* Exported functions */
38 void Function(void); // Normal
39
40 /* Private typedef */
41 /* Private define */
42 /* Private macro */
43 /* Private variables */
44 /* Private function prototypes */
45 void Function_Func1(void) __attribute__((section(".ARM._at_0x00003800"))); // Assign to IROM1 0x03800
46 void Function_Func2(void) __attribute__((section(".ARM._at_0x0001C000"))); // Assign to IROM1 0x1C00
47 void Function_Func3(void) __attribute__((section(".ARM._at_0x20000000"))); // Assign to IRAM1 Space
48 void Function_Func4(void) __attribute__((section(".ARM._at_0x20003800"))); // Assign to IRAM2 Space
49
50 /* Private user code */
51 /**
52 * @brief This function is executed in case of Function.
53 * @return None
54 */
55
56 void Function(void)
57 {
58     uint32_t lTemp32 = 0;
59     printf("...Function\n\r");
60
61     lTemp32 = 100;
62     do{
63

```

**Keyword : \_\_attribute\_\_((section(".ARM.\_at\_Address"))))**

**Note : Not support .c source file configuration to memory space.**

**Appendix**

1. Sample project : SampleProject\_ExecuteOnSRAM\_URT\_ShowChar.zip

**Revision History**

Revision V1.00 (2024_0105)		Chapter
1	Initial version	