Chapter 11
Interfacing C and Assembly Code

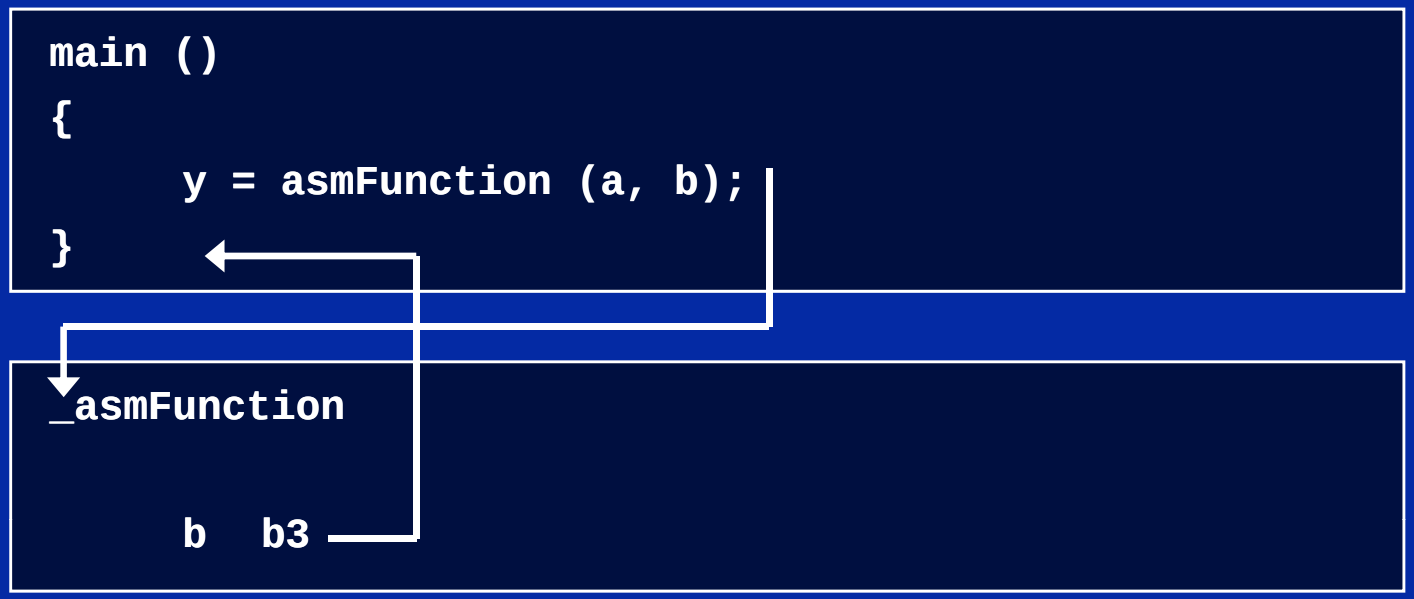
Learning Objectives

- Different methods exist for interfacing C and assembly code:
  - Calling assembly from C.
  - Interrupt calling assembly routine.
  - Intrinsics.
- Programming requirements when interfacing code.

Introduction

- This chapter shows how to interface C and assembly and how to use intrinsics.
- As a general rule the code written in C is used for initialisation and for non-critical (in terms of speed or size) code.
- Critical code (in terms of speed/size) can be written in assembly or linear assembly.
- There are three different ways to interface C and assembly code:
  - (1) C code call to the assembly function.
  - (2) An interrupt can call an assembly function.
  - (3) Call an assembly instruction using intrinsics.

Calling Assembly from C



- The C and assembly functions share the same resources (e.g. registers).
- The C and assembly functions may exchange data.
- Therefore code interfacing requires a means of handing-off data and control info and some rules of handling shared registers.

Calling Assembly from C

main_c.c

```
int asm_Function (short, short);

short x = 0x4000, y = 0x2000;
int z;

void main (void)
{
      z = asm_Function (x, y);
}
```

- Use “_” underscore in assembly for all variables or functions declared in C.
- Labels also need to be global.

asm_Function.c

```
int asm_Function (short a, short b)
{
      int y;
      y = (a * b) << 1;
      return y;
}
```

asm_Function.asm

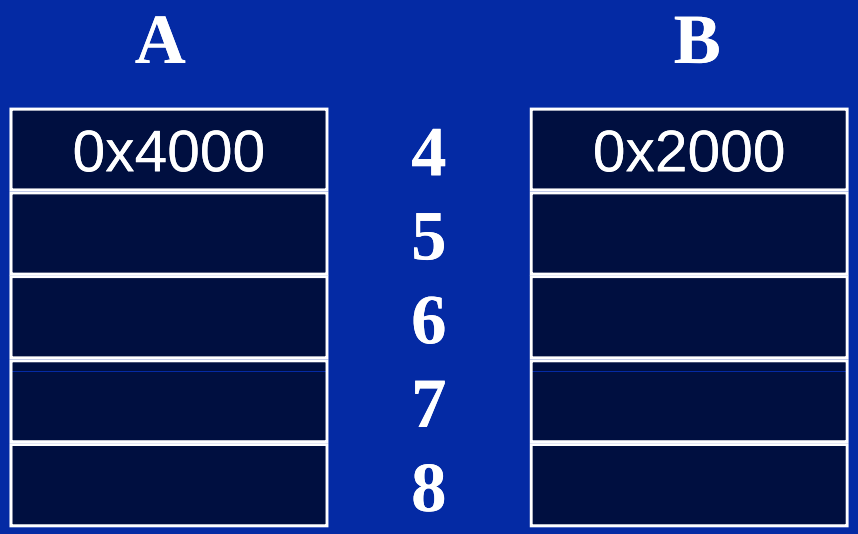
```
      .global _asm_Function
```

Passing Arguments between C and Assembly

- The following registers are used to pass and return variables when calling an assembly routine from C.

Passing Arguments between C and Assembly

- **Before assembly call.**



- **After return from assembly call.**

Passing Arguments between C and Assembly

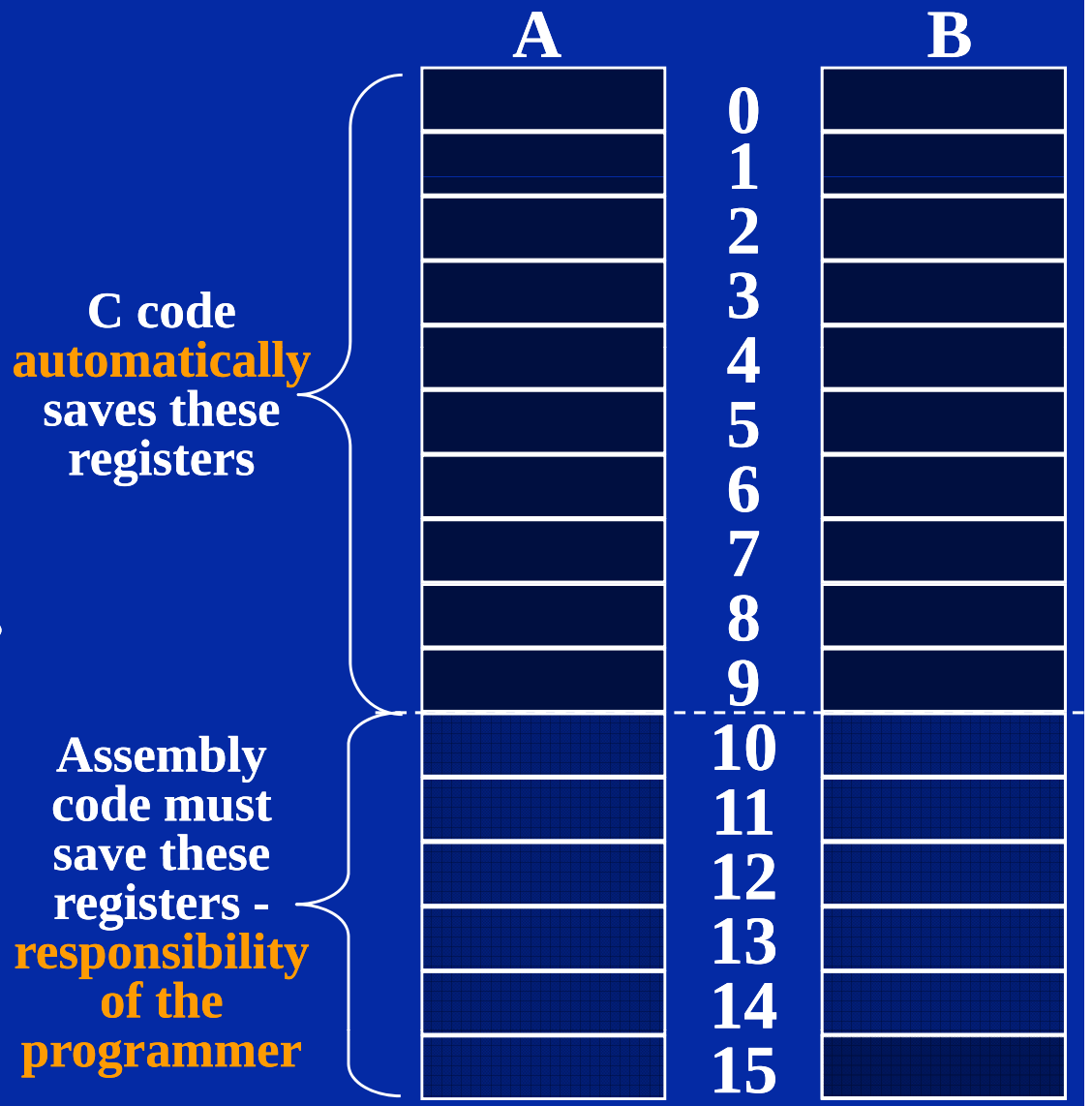
Problem:

- **The C code will use some or all of the registers.**
- **The assembly code may also require the use of some or all registers.**
- **If nothing is done then on return to the C code some of the values may have been destroyed by the assembly code.**

Passing Arguments between C and Assembly

Solution:

- Both the C code and assembly code are responsible for saving some registers if they need to use them.



Interfacing C and Assembly Examples

- Setup code written in C and interfaced with critical code written in assembly can be found in the following chapters:
  - \Code\Chapter 14 - Finite Impulse Response Filters
  - \Code\Chapter 16 - Adaptive Filters
- Setup code written in C and interfaced with critical code written in linear assembly can be found in the following chapters:
  - \Code\Chapter 15 - Infinite Impulse Response Filters
  - \Code\Chapter 17 - Goertzel Algorithm

Chapter 11
Interfacing C and Assembly Code
- End -