9.2 SUBROUTINE

A **subroutine** is a group of instructions written separately from the main program to perform a function that occurs repeatedly in the main program. For example, if a time delay is required between three successive events, three delays can be written in the main program. To avoid repetition of the same delay instructions, the subroutine technique is used. Delay instructions are written once, separately from the main program, and are called by the main program when needed.

The 8085 microprocessor has two instructions to implement subroutines: CALL (call a subroutine), and RET (return to main program from a subroutine). The CALL instruction is used in the main program to call a subroutine, and the RET instruction is used at the end of the subroutine to return to the main program. When a subroutine is called, the contents of the program counter, which is the address of the instruction following the CALL instruction, is stored on the stack and the program execution is transferred to the subroutine address. When the RET instruction is executed at the end of the subroutine, the memory address stored on the stack is retrieved, and the sequence of execution is resumed in the main program. This sequence of events is illustrated in Example 9.3.

INSTRUCTIONS

Opcode	Operand	
CALL	16-bit memory address of a subroutine	Call Subroutine Unconditionally ☐ This is a 3-byte instruction that transfers the program sequence to a subroutine address ☐ Saves the contents of the program counter (the address of the next instruction) on the stack ☐ Decrements the stack pointer register by two ☐ Jumps unconditionally to the memory location specified by the second and third bytes. The second byte specifies a line number and the third byte specifies a page number ☐ This instruction is accompanied by a return instruction in the subroutine
RET		Return from Subroutine Unconditionally ☐ This is a 1-byte instruction ☐ Inserts the two bytes from the top of the stack into the program counter and increments the stack pointer register by two ☐ Unconditionally returns from a subroutine

The conditional Call and Return instructions will be described later in the chapter.

Example 9.3 Illustrate the exchange of information between the stack and the program counter for the following program if the available user memory ranges from 2000H to 23FFH.

```
Memory
Address
2000        LXI SP,2400H        ;Initialize the stack pointer at 2400H
  ↓           ↓
2040        CALL 2070H          ;Call the subroutine located at 2070H. This is
2041                            ;   a 3-byte instruction
2042
2043        NEXT INSTRUCTION    ;The address of the next instruction following
  |           |                 ;   the CALL instruction
  ↓           ↓
205F        HLT                 ;End of the main program
2070        First Subroutine    ;Beginning of the subroutine
  |           Instruction
  ↓           ↓
207F        RET                 ;End of the subroutine
2080        ↓
  |         Other Subroutines
  ↓         ↓
2398        Empty Space
23FF        ↓
2400                            ;The stack is initialized at 2400H
```

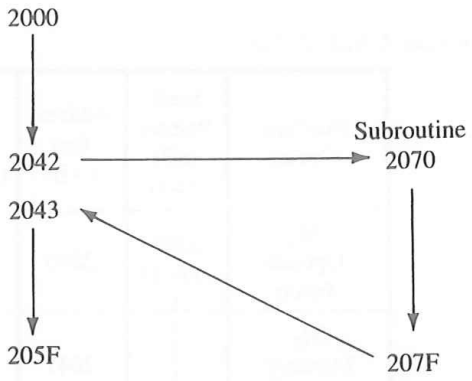
Solution After reviewing the above program note the following points:

1. The available user memory is from 2000H to 23FFH (1024 or 1K bytes); however, the stack pointer register is initialized at 2400H, one location beyond the user memory. This allows maximum use of the memory because the actual stack begins at 23FFH. The stack can expand up to the location 2398H without overlapping with the program.
2. The main program is stored at memory locations from 2000H to 205FH.
3. The CALL instruction is located at 2040H to 2042H (3-byte instruction). The next instruction is at 2043H.
4. The subroutine begins at the address 2070H and ends at 207FH.

PROGRAM EXECUTION

The sequence of the program execution and the events in the execution of the CALL and subroutine are shown in Figures 9.9 and 9.10.

FIGURE 9.9
Subroutine Call and Program Transfer



The program execution begins at 2000_H, continues until the end of CALL 2042_H, and transfers to the subroutine at 2070_H. At the end of he subroutine, after executing the RET instruction, it comes back to the main program at 2043H and continues.

CALL EXECUTION

Memory Address	Machine Code	Mnemonics	Comments
2040	CD	CALL 2070H	;Call subroutine located at the memory
2041	70		; location 2070H
2042	20		
2043	NEXT	INSTRUCTION	

The sequence of events in the execution of the CALL instruction by the 8085 is shown in Figure 9.10. The instruction requires five machine cycles and eighteen T-states. The sequence of events in each machine cycle is as follows.

1. M_1—Opcode Fetch: In this machine cycle, the contents of the program counter (2040H) are placed on the address bus, and the instruction code CD is fetched using the data bus. At the same time, the program counter is upgraded to the next memory address, 2041H. After the instruction is decoded and executed, the stack pointer register is decremented by one to 23FFH.
2. M_2 and M_3—Memory Read: These are two Memory Read cycles during which the 16-bit address (2070H) of the CALL instruction is fetched. The low-order address 70H is fetched first and placed in the internal register Z. The high-order address 20H is fetched next, and placed in register W. During M_3, the program counter is upgraded to 2043H, pointing to the next instruction.
3. M_4 and M_5—Storing of Program Counter: At the beginning of the M_4 cycle, the normal operation of placing the contents of the program counter on the address bus is suspended; instead, the contents of the stack pointer register 23FFH are placed on the address bus. The high-order byte of the program counter (PCH = 20H) is placed on the data bus and stored in the stack location 23FFH. At the same time, the stack pointer register is decremented to 23FEH.

   During machine cycle M_5, the contents of the stack pointer 23FEH are placed on the address bus. The low-order byte of the program counter (PCL = 43H) is placed on the data bus and stored in stack location 23FEH.

Instruction: CALL 2070H

Memory Address	Code (H)
2040	CD
2041	70
2042	20

Machine Cycles	Stack Pointer (SP) 2400	Address Bus (AB)	Program Counter (PCH) (PCL)	Data Bus (DB)	Internal Registers (W) (Z)
M_1 Opcode Fetch	23FF (SP–1)	2040	20 41	CD Opcode	—
M_2 Memory Read		2041	20 42	70 Operand	→ 70
M_3 Memory Read	23FF	2042	20 43	20 Operand	→ 20
M_4 Memory Write	23FE (SP–2)	23FF	20 43	20 → (PCH)	
M_5 Memory Write	23FE	23FE	20 43	43 → (PCL)	(20) (70)
M_1 Opcode Fetch of Next Instruction		2070 → (W)(Z)	2071		(2070) (W)(Z)

FIGURE 9.10
Data Transfer During the Execution of the CALL Instruction

4. Next Instruction Cycle: In the next instruction cycle, the program execution sequence is transferred to the CALL location 2070H by placing the contents of W and Z registers (2070H) on the address bus. During M_1 of the next instruction cycle, the program counter is upgraded to location 2071 (W,Z + 1).

In summary: After the CALL instruction is fetched, the 16-bit address (the operand) is read during M_2 and M_3 and stored temporarily in W/Z registers. (Examine the contents of the address bus and the data bus in Figure 9.10.) In the next two cycles, the contents of the program counter are stored on the stack. This is the address where the microprocessor will continue the execution of the program after the completion of the subroutine. Figure 9.11 shows the contents of the program counter, the stack pointer register, and the stack during the execution of the CALL instruction.

RET EXECUTION

At the end of the subroutine, when the instruction RET is executed, the program execution sequence is transferred to the memory location 2043H. The address 2043H was

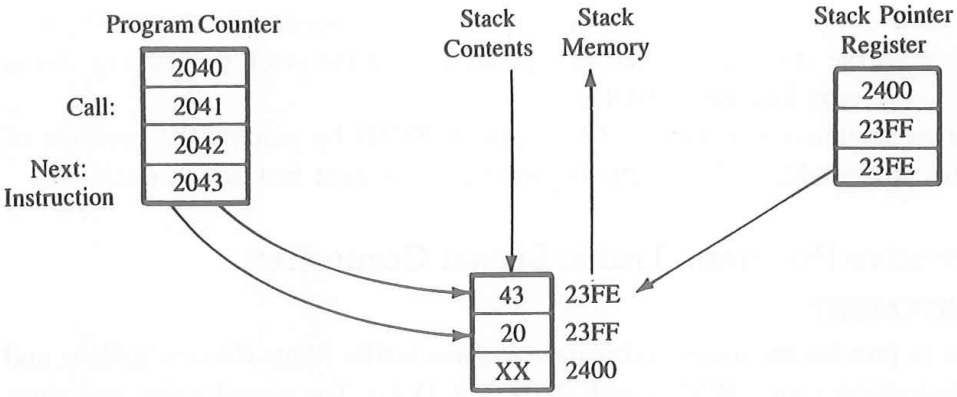


FIGURE 9.11
Contents of the Program Counter, the Stack Pointer, and the Stack During the Execution of the CALL Instruction

stored in the top two locations of the stack (23FEH and 23FFH) during the CALL instruction. Figure 9.12 shows the sequence of events that occurs as the instruction RET is executed.

M_1 is a normal Opcode Fetch cycle. However, during M_2 the contents of the stack pointer register are placed on the address bus, rather than those of the program counter. Data byte 43H from the top of the stack is fetched and stored in the Z register, and the

Instruction: RET

Memory Address	Code (H)
207F	C9

Contents of Stack Memory	
23FE	43
23FF	20

Machine Cycles	Stack Pointer (23FE)	Address Bus (AB)	Program Counter	Data Bus (DB)	Internal Registers (W) (Z)
M_1 Opcode Fetch	23FE	207F	2080	C9 Opcode	
M_2 Memory Read	23FF	23FE		43 (Stack) →	→ 43
M_3 Memory Read	2400	23FF		20 (Stack–1) →	→ 20
M_1 Opcode Fetch of Next Instruction		2043 (W) (Z) ←	2044		2043 (W) (Z) ←

FIGURE 9.12
Data Transfer During the Execution of the RET Instruction