Exercise 3: Data Movement between General Purpose Registers

Objective

To practice move instructions involving data transfer and basic CPU architecture. The data transfer involves register-to-register, immediate-data-to-register, memory-to-register, register-to-memory, and immediate-data-to-memory. Create in-line assembly programs for data moving between the CPU and RAM. Exam the functions and contents of general registers of AL, AH, AX, and EAX; BL, BH, BX, and EBX; CL, CH, CX, and ECX; DL, DH, DX, and EDX. The examples involve MOV instructions: MOV EAX, EBX; MOV AX, 0xA234; MOV AH, 15; MOV AL, 'A'.

Type and understand Example 1, Example 2, and Example 3 listed in the exercises. Compile, build, and run this program. Understand the structure of in-line assembly programming. Explain each line of the program and describe the output of each in your lab report.

1. Write an in-line assembly language module in a C program to explore BL, BH, BX, respectively.
   (a) MOV ten char 'A', 'B', … 'J', 'a', 'b', …'j' into 8-bit registers, respectively. Upper case letter goes BH, lower case goes BL
   (b) Swap the information stored at BH and BL general registers.
   (c) Show the contents of BX, BH, and BL general registers before and after swapping.
   (d) Find the hexadecimal value ASCII code characters of 'A', 'B', … 'J', and 'a', 'b', … 'j'.
2. Write an in-line assembly language module in a C program to explore AL, AH, AX, respectively.
   (a) MOV ten char 'A', 'B', … 'J', 'a', 'b', …'j' into 8-bit registers, respectively. Upper case letter goes AH, lower case goes AL
   (b) Swap the information stored at AH and AL general registers.
   (c) Show the contents of AX, AH, and AL general registers before and after swapping.
   (d) Continue your ASM programming and explore more general registers CX/DX, CH/DH and CL/DL. Show all register contents before and after swapping.
3. For Example 3 (SwapXBytesErr.cpp), find the logic error inside second _asm{} and make necessary changes.
   You report should include parts of 1, 2, and 3. You also need to include all source codes for examples 1, 2, and 3. Include the program output in your lab report. If you have any syntax errors in your code, include a printout of the incorrect code with written annotations describing how you fixed the problem(s) in your program.

Written exercises:

- A title page with the lab title, your name and other identification, and the current date
- If you modify or create a program in an exercise, the source code with comments for that program should appear in the laboratory report.
- The output from all programs should also appear in the lab report.
- For each exercise, you should provide a write-up describing the purpose of the exercise, what you learned from the exercise, and any comments, improvements, or other work you've done with the exercise.

printf("BH=%c, BL=%c, BX=%c%c\n" r2, r1, r2, r1);
AH AL BX *

Example 3.1 SwapBytes.cpp

```
/*************************************************************
File name: SwapBytes.cpp
Swap bytes in two general registers and exam the functions and contents
Method: Embedding an in-line assembly language module into a C program

A. Swap bytes in two general registers of BL, BH
B. Swap bytes in two general registers of AL, AH

printf() is used to display information on the console
scanf_s() is used to receive the integers from the keyboard
*************************************************************/
#include <stdio.h>      // to use printf()
#include <iostream>     // to use system ("pause") to  hold  the console;

int main()
{
   char temp;
   char r1, r2;
   short int r;
            ASSIGNING DATA
   //A. Swap bytes in two general registers of BL, BH
   _asm
   {
      MOV BL, 'a';
      MOV BH, 'A';
      MOV r1, BL;
      MOV r2, BH;
      MOV r, BX;
   }
   std::cout << "========================================== " << std::endl;
   std::cout << "Before swapping " << std::endl;
   // to display characters
   printf ("BH = %c, BL = %c\n", r2, r1);
   // to display hexidecimal value of each character
   printf ("BH =0x%x, BL = 0x%x, BX = 0x%x\n", r2, r1, r);
            SWAPPING DATA
   //Swap bytes in two general registers of BL, BH
   _asm
   {
      MOV temp, BH;
      MOV BH,  BL;
      MOV BL, temp;
      MOV r1, BL;
      MOV r2, BH;
      MOV r, BX;
   }
   std::cout << "========================================== " << std::endl;
   std::cout << "After swapping " << "\n";
```

BX = BHBL → a, A } 8-bit
AX = AHAL → b, B

16-bit { c, C
d, D

32-bit { e, E
F, E

characters

BL = a
BH = A
r1 = a | X
r2 = A
BX = Aa

BX = BHBL

a, A

BX = %c%c } Missing *
r2, r1)
BX = BHBL

temp = A
BH = a
BL = A
r1 = a
r2 = A
r = aA

a, A

swapped

```
// to display characters
printf ("BH = %c, BL =%c \n", r2, r1);
// to display hexadecimal value of each character
printf ("BH = 0x%x, BL =0x%x , BX= 0x%x \n", r2, r1, r);
/*************************************************************************/
        ASSIGNING DATA
//B. Swap bytes in two general registers of AL, AH
_asm
{
    MOV AL, 'b';
    MOV AH, 'B';
    MOV r1, AL;
    MOV r2, AH;
    MOV r, AX;
}
std::cout << "========================================" << std::endl;
std::cout << "Before swapping " << std::endl;
// to display characters
printf ("AH = %c, AL = %c\n", r2, r1);
// to display hexadecimal value of each character
printf ("AH =0x%x, AL = 0x%x, AX = 0x%x\n", r2, r1, r);
        SWAPPING DATA
//Swap bytes in two general registers AL and AH
_asm
{
    MOV AL, 'b';
    MOV AH, 'B';
    MOV temp, AH;
    MOV AH,  AL;
    MOV AL, temp;
    MOV r1, AL;
    MOV r2, AH;
    MOV r, AX;
}
std::cout << "========================================" << std::endl;
std::cout << "After swapping " << "\n";

// to display characters
printf ("AH = %c, AL =%c \n", r2, r1);
// to display hexadecimal value of each character
printf ("AH = 0x%x, AL =0x%x , AX= 0x%x \n", r2, r1, r);

system ("pause");
return 0;
}
```

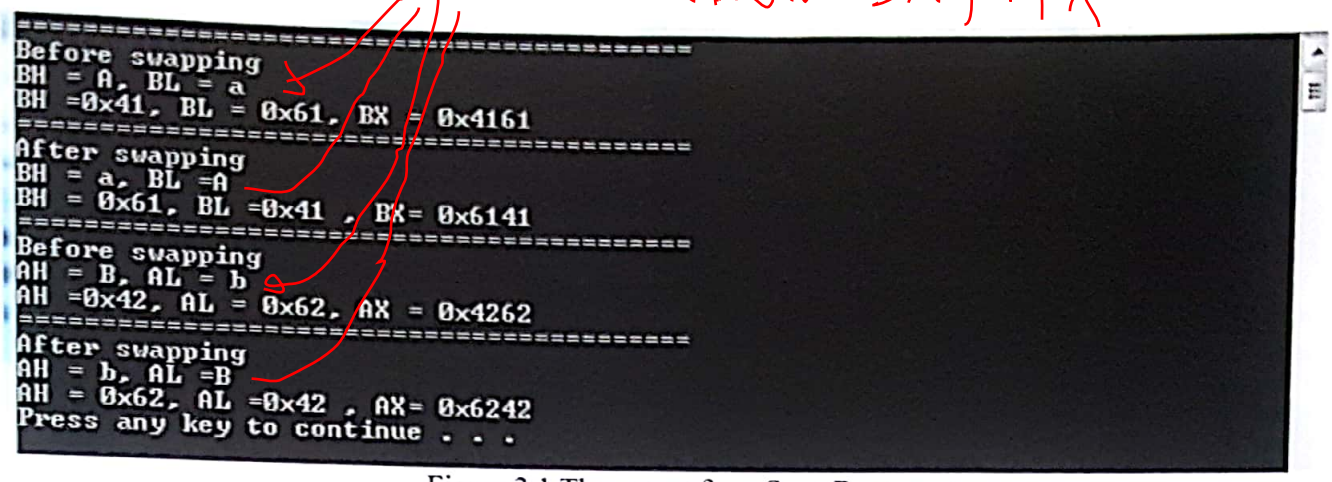

Figure 3.1 The output from SwapBytes.cpp

Example 3.2 SwapXBytes.cpp

```
/*************************************************************
File name: SwapXBytes.cpp
Swap bytes in two general registers and exam the functions and contents
Method: Embedding an in-line assembly language module into a C program

A. Swap bytes in two general registers BL, BH
B. Swap bytes in two general registers AL, AH

printf() function and scanf_s() functions:
printf() is used to display information on the console
scanf_s() is used to receive the integers from the keyboard
std::cout is used to display information on the console
system ("pause") is used to hold the console
XCHG instruction is used to swap bytes, for example, XCHG (XCHG BL, BH)
*************************************************************/
#include <stdio.h>      // to use printf()
#include <iostream>     // to use system ("pause") to  hold  the console;
int main()
{
    char r1, r2;
    short int r;
    //A. Swap bytes in two general registers BL, BH
    _asm
    {
        MOV BL, 'i';
        MOV BH, 'I';
        MOV r1, BL;
        MOV r2, BH;
        MOV r, BX;
    }
    std::cout << "========================================= " << std::endl;
    std::cout << "Before swapping " << std::endl;
```

```
// To display characters
printf ("BH = %c, BL = %c\n", r2, r1);

// Display the hexadecimal value of each character
printf ("BH =0x%x, BL = 0x%x, BX = 0x%x\n", r2, r1, r);

// Swap bytes in two general registers BL, BH
_asm
    {
    XCHG BL, BH;
    MOV r1, BL;
    MOV r2, BH;
    MOV r, BX;
}
std::cout << "========================================== " << std::endl;
std::cout << "After swapping " << "\n";
// To display characters
printf ("BH = %c, BL =%c \n", r2, r1);

// Display the hexadecimal value of each character
printf ("BH = 0x%x, BL =0x%x , BX= 0x%x \n", r2, r1, r);
/*****************************************************************/
//B. Swap bytes in two general registers AL, AH
_asm
{
    MOV AL, 'j';
    MOV AH, 'J';
    MOV r1, AL;
    MOV r2, AH;
    MOV r, AX;
}
std::cout << "========================================== " << std::endl;
std::cout << "Before swapping " << std::endl;

// To display characters
printf ("AH = %c, AL = %c\n", r2, r1);
// Display the hexadecimal value of each character
printf ("AH =0x%x, AL = 0x%x, AX = 0x%x\n", r2, r1, r);

// Swap bytes in two general registers AL, AH
_asm{
    MOV AL, 'j';
    MOV AH, 'J';
    XCHG AL, AH;
    MOV r1, AL;
    MOV r2, AH;
    MOV r, AX;
}
```

```
    std::cout << "================================================== " << std::endl;
    std::cout << "After swapping " << "\n";

    // to display characters
    printf ("AH = %c, AL =%c \n", r2, r1);

    // Display the hexadecimal value of each character
    printf ("AH = 0x%x, AL =0x%x , AX= 0x%x \n", r2, r1, r);

    system ("pause");
    return 0;
}
```

AX missing

BX, AX are missing

```
==================================================
Before swapping
BH = I, BL = i
BH =0x49, BL = 0x69, BX = 0x4969
==================================================
After swapping
BH = i, BL =I
BH = 0x69, BL =0x49 , BX= 0x6949
==================================================
Before swapping
AH = J, AL = j
AH =0x4a, AL = 0x6a, AX = 0x4a6a
==================================================
After swapping
AH = j, AL =J
AH = 0x6a, AL =0x4a , AX= 0x6a4a
Press any key to continue . . .
```

Figure 3.2 The output from SwapXBytes.cpp

Example 3.3 SwapXBytesErr.cpp → for Laboratory No 2

```
/*************************************************************
file name: SwapXBytesErr.cpp
Swap bytes in two general registers and exam the functions and contents
Method: Embedding an in-line assembly language module into a C program

A. Swap bytes in two general registers BL, BH
B. Swap bytes in two general registers AL, AH

printf() function and scanf_s() functions:
printf() is used to display information on the console
scanf_s() is used to receive the integers from the keyboard
std::cout is used to display information on the console
system ("pause") is used to hold the console
XCHG instruction is used to swap bytes, for example, XCHG (XCHG BL, BH)
Find the logic error inside asm{} and make necessary changes.
*************************************************************/
#include <stdio.h>  // to use printf()
#include <iostream>  // to use system ("pause") to  hold  the console;
```

USE TWO EXCHANGE 'XCHG' INSTRUCTIONS

THE DEAL IS WITH THE SECOND XCHG TO BE CORRECTED.

ADD THE 'temp' INSTRUCTION TO EXAMPLE 3.3 FROM EXAMPLE 3.1 AND SEE HOW THE FIRST XCHG WORKS.

```
int main()
{
    char r1, r2;
    short int r;
    //A. Swap bytes in two general registers BL, BH
    _asm
    {
        MOV BL, 'i';
        MOV BH, 'I';
        MOV r1, BL;
        MOV r2, BH;
        MOV r, BX;
    }
    std::cout << "========================================= " << std::endl;
    std::cout << "Before swapping " << std::endl;

    // To display characters
    printf ("BH = %c, BL = %c\n", r2, r1);
    // Display the hexadecimal value of each character
    printf ("BH =0x%x, BL = 0x%x, BX = 0x%x\n", r2, r1, r);

    // Swap bytes in two general registers BL, BH
    _asm
    {
        XCHG BL, BH;
        MOV r1, BL;
        MOV r2, BH;
        MOV r, BX;
    }
    std::cout << "========================================= " << std::endl;
    std::cout << "After swapping " << "\n";

    // To display characters
    printf ("BH = %c, BL =%c \n", r2, r1);

    // Display the hexadecimal value of each character
    printf ("BH = 0x%x, BL =0x%x , BX= 0x%x \n", r2, r1, r);
    /*****************************************************************/
    //B. Swap bytes in two general registers AL, AH
    _asm
    {
        MOV AL, 'j';
        MOV AH, 'J';
        MOV r1, AL;
        MOV r2, AH;
        MOV r, AX;
    }
    std::cout << "========================================= " << std::endl;
    std::cout << "Before swapping " << std::endl;
```

Use temp from example 3.1 add

Assigning

BX missing

BX missing

Assigning

```
// To display characters
printf ("AH = %c, AL = %c\n", r2, r1);
// Display the hexadecimal value of each character
printf ("AH =0x%x, AL = 0x%x, AX = 0x%x\n", r2, r1, r);

// Swap bytes in two general registers AL, AH
_asm
{
   XCHG AL, AH;
   MOV r1, AL;
   MOV r2, AH;
   MOV r, AX;
}
std::cout << "========================================= " << std::endl;
std::cout << "After swapping " << "\n";

// to display characters
printf ("AH = %c, AL =%c \n", r2, r1);

// Display the hexadecimal value of each character
printf ("AH = 0x%x, AL =0x%x , AX= 0x%x \n", r2, r1, r);

system ("pause");
return 0;
}
```

AX missing

for student to find

NEED TO BE CORRECTED

The output is not right

AX missing

AX = AH + AL

BX = BH + BL

AX, BX are missing

```
=========================================
Before swapping
BH = I, BL = i
BH =0x49, BL = 0x69, BX = 0x4969
=========================================
After swapping
BH = i, BL =I
BH = 0x69, BL =0x49 , BX= 0x6949
=========================================
Before swapping
AH = J, AL = j
AH =0x4a, AL = 0x6a, AX = 0x4a6a
=========================================
After swapping
AH =!, AL =
AH = 0x21, AL =0x0 , AX= 0x2100
Press any key to continue . . .
```

Figure 3.3 The output from SwapXBytesErr.cpp

1. The students must find the missing AX and BX in the output result, then replace the letters a and b instead of i, j in example 3.3.
2. Add the exchange instruction by using the temporal 'temp' memory as illustrated in example 3.1.
2. The students must correct the output result in example 3.3 and use letters a and b instead.
3. Work on your report by showing code, code explanation, flow chart,output results, comements, and conclusion with real application for 30 points.