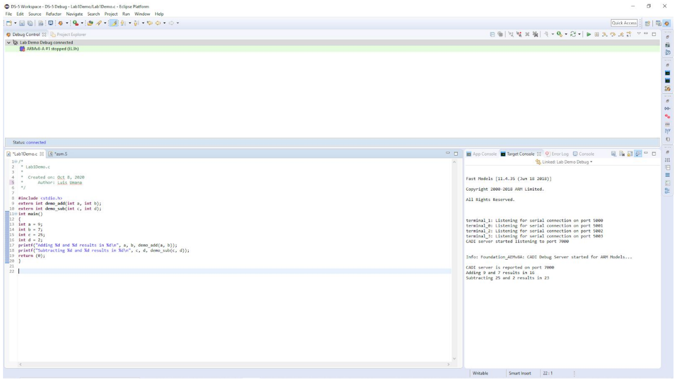
Lab 1 A:

Explain the following in detail with respect to the c code and assembly provided:

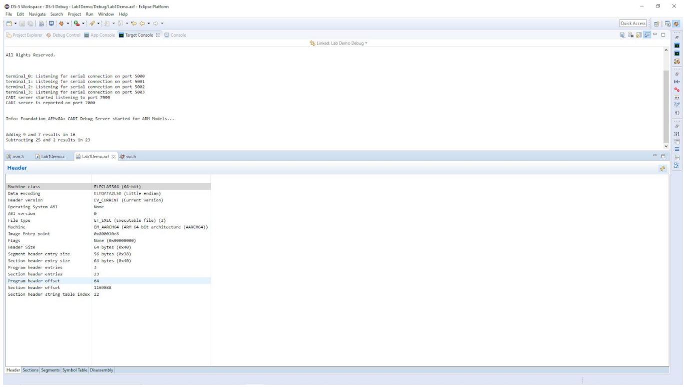
For this lab, the C code initializes the variables, A-D, contains two external functions and prints out two statements. The first statement is adding variable A and B and the second statement is subtracting variable C and D. The external functions for adding and subtracting are stated in the C code so that they can be used in assembly. The assembly code first adds variable A and B, in this case X0 and X1, and then the branch statement resets the registers. Once the registers are resetted, the program then subtracts the new X0 and X1, variable C and D.

Demo program screenshots (building and executing in debug window, displaying the output along with project explorer tab)



```c
/*
 * Lab1Demo.c
 *
 *  Created on: Oct 8, 2020
 *      Author: Luis Umana
 */

#include <stdio.h>
extern int demo_add(int a, int b);
extern int demo_sub(int c, int d);
int main()
{
int a = 9;
int b = 7;
int c = 25;
int d = 2;
printf("Adding %d and %d results in %d\n", a, b, demo_add(a, b));
printf("Subtracting %d and %d results in %d\n", c, d, demo_sub(c, d));
return (0);
}
```

```
Fast Models [11.4.35 (Jun 18 2018)]

Copyright 2000-2018 ARM Limited.

All Rights Reserved.


terminal_1: Listening for serial connection on port 5000
terminal_0: Listening for serial connection on port 5001
terminal_2: Listening for serial connection on port 5002
terminal_3: Listening for serial connection on port 5003
CADI server started listening to port 7000


Info: Foundation_AEMv8A: CADI Debug Server started for ARM Models...

CADI server is reported on port 7000
Adding 9 and 7 results in 16
Subtracting 25 and 2 results in 23
```



```
All Rights Reserved.


terminal_0: Listening for serial connection on port 5000
terminal_1: Listening for serial connection on port 5001
terminal_2: Listening for serial connection on port 5002
terminal_3: Listening for serial connection on port 5003
CADI server started listening to port 7000
CADI server is reported on port 7000


Info: Foundation_AEMv8A: CADI Debug Server started for ARM Models...


Adding 9 and 7 results in 16
Subtracting 25 and 2 results in 23
```

Machine class	ELFCLASS64 (64-bit)
Data encoding	ELFDATA2LSB (Little endian)
Header version	EV_CURRENT (Current version)
Operating System ABI	None
ABI version	0
File type	ET_EXEC (Executable file) (2)
Machine	EM_AARCH64 (ARM 64-bit architecture (AARCH64))
Image Entry point	0x800010e8
Flags	None (0x00000000)
Header Size	64 bytes (0x40)
Segment header entry size	56 bytes (0x38)
Section header entry size	64 bytes (0x40)
Program header entries	3
Section header entries	23
Program header offset	64
Section header offset	1169088
Section header string table index	22

Explain in detail about the .axf file generated along with the Header, Sections, Segments, Symbol Table, Disassembly details.

The only way to get a result from the program is by debugging and running the .axf file. The header provides details of the file itself, the entry sizes, how many flags were used, class, etc. The sections and segments provide the names of everything used in the program and it tells the address, flags and size. The symbol table provides the address, type and size.

Elaborate the Disassembly tab details from start up to the finish function.

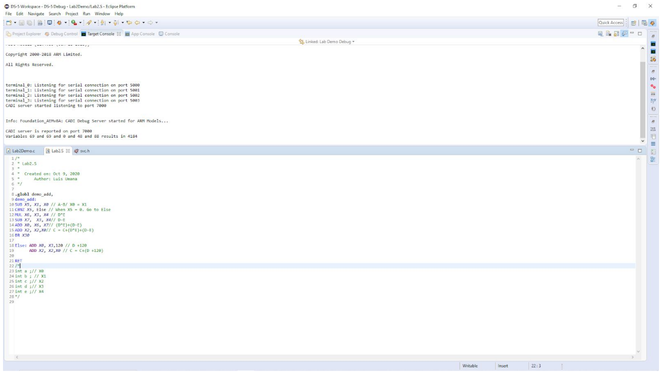
Initially, every value of A-Cis stored at w0 until the main begins to shift itself so that B ends up at w1 and C at w2. Once every number is in their respective position, the program then beings to execute the assembly code

Provide the details of different variables and registers that are used by the program by making use of the debug control along with the start address and end address of each function.

Since there are 4 variables required for this part of the lab, the registers used for the program would be X0 - X1 in this case, because the branch X30 “clears out the registers” for the next part of the code to be executed.

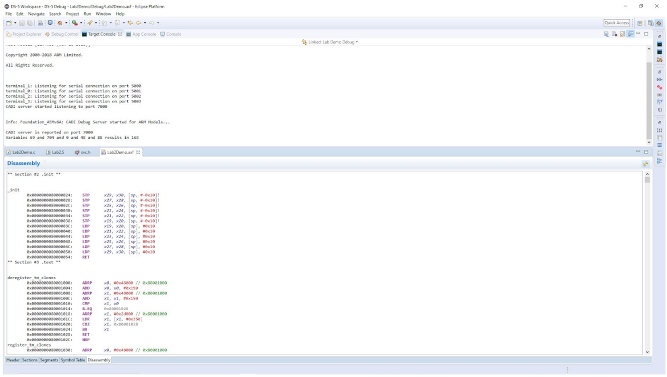
Lab 1 B:

C program and assembly program for question 2 along with screenshots (building and executing in debug window displaying the output along with project explorer tab).


DS-5 Workspace - DS-5 Debug - Lab2Demo/Lab2.S - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Project Explorer
Debug Control
Target Console
App Console
Console
Linked: Lab Demo Debug
Lab2Demo.c
Lab2.S
svc.h

```
Copyright 2000-2018 ARM Limited.

All Rights Reserved.


terminal_0: Listening for serial connection on port 5000
terminal_1: Listening for serial connection on port 5001
terminal_2: Listening for serial connection on port 5002
terminal_3: Listening for serial connection on port 5003
CADI server started listening to port 7000


Info: Foundation_AEMv8A: CADI Debug Server started for ARM Models...

CADI server is reported on port 7000
Variables 69 and 69 and 0 and 48 and 88 results in 4184
```

```
/*
 * Lab2.S
 *
 *  Created on: Oct 9, 2020
 *      Author: Luis Umana
 */

.globl demo_add,
demo_add:
SUB X5, X1, X0 // A-B/ X0 = X1
CBNZ X5, Else // When X5 = 0. Go to Else
MUL X6, X3, X4 // D*E
SUB X7,  X3, X4// D-E
ADD X0, X6, X7// (D*E)+(D-E)
ADD X2, X2,X0// C = C+(D*E)+(D-E)
BR X30

Else: ADD X0, X3,120 // D +120
      ADD X2, X2,X0 // C = C+(D +120)

RET
/*
int a ;// X0
int b ; // X1
int c ;// X2
int d ;// X3
int e ;// X4
*/
```



```
Copyright 2000-2018 ARM Limited.

All Rights Reserved.


terminal_1: Listening for serial connection on port 5000
terminal_0: Listening for serial connection on port 5001
terminal_2: Listening for serial connection on port 5002
terminal_3: Listening for serial connection on port 5003
CADI server started listening to port 7000


Info: Foundation_AEMv8A: CADI Debug Server started for ARM Models...

CADI server is reported on port 7000
Variables 69 and 704 and 0 and 48 and 88 results in 168
```

```
** Section #2 .init **


_init
        0x0000000080000024:    STP      x29, x30, [sp, #-0x10]!
        0x0000000080000028:    STP      x27, x28, [sp, #-0x10]!
        0x000000008000002C:    STP      x25, x26, [sp, #-0x10]!
        0x0000000080000030:    STP      x23, x24, [sp, #-0x10]!
        0x0000000080000034:    STP      x21, x22, [sp, #-0x10]!
        0x0000000080000038:    STP      x19, x20, [sp, #-0x10]!
        0x000000008000003C:    LDP      x19, x20, [sp], #0x10
        0x0000000080000040:    LDP      x21, x22, [sp], #0x10
        0x0000000080000044:    LDP      x23, x24, [sp], #0x10
        0x0000000080000048:    LDP      x25, x26, [sp], #0x10
        0x000000008000004C:    LDP      x27, x28, [sp], #0x10
        0x0000000080000050:    LDP      x29, x30, [sp], #0x10
        0x0000000080000054:    RET
** Section #3 .text **


deregister_tm_clones
        0x0000000080001000:    ADRP     x0, #0x48000 // 0x80001000
        0x0000000080001004:    ADD      x0, x0, #0x150
        0x0000000080001008:    ADRP     x1, #0x48000 // 0x80001000
        0x000000008000100C:    ADD      x1, x1, #0x150
        0x0000000080001010:    CMP      x1, x0
        0x0000000080001014:    B.EQ     0x80001028
        0x0000000080001018:    ADRP     x1, #0x2d000 // 0x80001000
        0x000000008000101C:    LDR      x1, [x1, #0x350]
        0x0000000080001020:    CBZ      x1, 0x80001028
        0x0000000080001024:    BR       x1
        0x0000000080001028:    RET
        0x000000008000102C:    NOP
register_tm_clones
        0x0000000080001030:    ADRP     x0, #0x48000 // 0x80001000
```

```
/*Lab2Demo.c*/
#include <stdio.h>
extern int demo_add(int a, int b, int c, int d, int e);
int main()
{
int a = 69;
int b = 704;
int c = 0;
int d = 48;
int e = 88;
printf("Variables %d and %d and %d and %d and %d results in %d\n", a, b, c, d, e,
demo_add(a, b, c, d, e));

return (0);
}
```

```
/*Lab2.S*/

.globl demo_add,
demo_add:
SUB X5, X1, X0 // A-B/ X0 = X1
CBNZ X5, Else // When X5 = 0. Go to Else
MUL X6, X3, X4 // D*E
SUB X7,  X3, X4// D-E
ADD X0, X6, X7// (D*E)+(D-E)
ADD X2, X2,X0// C = C+(D*E)+(D-E)
BR X30

Else: ADD X0, X3,120 // D +120
     ADD X2, X2,X0 // C = C+(D +120)

RET
/*
int a ;// X0
int b ; // X1
int c ;// X2
int d ;// X3
int e ;// X4
*/
```

Explain the following in detail with respect to the c code and assembly implemented:

The C code contains 5 variables, one external function, and one print system. The C code's only purpose in this case is to set up the external function and initialize the required variables. The assembly code starts by subtracting the variable A and B and storing the value in X5. If the value of X5 is equal to 0 then it will go to the Else statement, however if it does not equal 0 then it will execute the first statement. When X5 is equal to 0, the program will add the sum of D multiply E and D minus E. When the X5 is not equal to 0, the else statement will be executed which will be the sum of D and 120.

Explain in detail about the .axf file generated along with the Header, Sections, Segments, Symbol Table, Disassembly details.

The only way to get a result from the program is by debugging and running the .axf file. The header provides details of the file itself, the entry sizes, how many flags were used, class, etc. The sections and segments provide the names of everything used in the program and it tells the address, flags and size. The symbol table provides the address, type and size.

Elaborate the Disassembly tab details from start up to the finish function.

Initially, every value of A-Cis stored at w0 until the main begins to shift itself so that B ends up at w1 and C at w2. In the middle of the disassembly , there is a BEQ statement which shows that the branch if statement is working in the assembly code. The rest of the statements proceed to intimate the code written in the assembly file.

Provide the details of different variables and registers that are used by the program by making use of the debug control along with the start address and end address of each function.

There are 5 variables required for this part of the lab, the registers used for the program are X0 - X7. Register X5 is used to validate that A and B are the same value by subtracting them and seeing whether the final value is 0 or something else. If the value is 0, this means that the values of A and B are the same and the first part of the program will be executed only. X6 is the value of D times E. X7 is the value of D minus E. X0 is then used to obtain the sum of X6 and X7. Since we are solving for C, aka X2, X0 is added to X2 to obtain the answer for C. If the value is not 0 then the 120 would be added to the value of X3 and stored into X0. X0 is then added to X2 to get our final answer for C.