Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 9, August 2021: 8027-8033

A Study on Determining the Link between the C Programming and Microprocessor

Helaria Maria
Assistant Professor, New Horizon College, Bangalore, India
helariax@gmail.com

ABSTRACT

A basic and very well-known programming is C program or high-level language and Microprocessor a Assemble Language Programming. C is translated to assembly language, but how its eventually gets executed on a microprocessor. A physicalunderstanding of how a C program is executed in a microprocessor and must of the people do not understand what exactly happens in various segments of the memory,when a program is being executed, so we are bridging a gap between these two languages. Where are the local variables stored in C? we quickly answer its stored in stack. What is kind of scope do, variables in a function have, there is no understanding on these things. In this paper we ensure the understanding between two languages exactly what happens when a C program is executed in the microprocessor.

INTRODUCTION

C is a general-purpose programming language, and is used for writing programs in many different domains, such as operating systems, numerical computing, graphical applications, etc. It is a small language, with just 32 keywords [1]. A microprocessor can simply be defined as a black box that can do certain computations so, we call this asMup and this is also called as random access memory, a RAM (Random Access Memory) So, a microprocessor hasdata bus, that lines here indicates that this could be more than one bit it could typically be 8 bits, 16 bits, 32, 64. The data bus is actually going to be bidirectional means, microprocessor can send data to the memoryor it can receive data from the memory. The address lines depending on the size of the memory, and you know how much memory it can access, it is that many bits will be assigned. Typically, it is going to be 2^n, if n is the number of address bits, then the number of locations that it can logically address is 2^n. The memory also has two other control signals called the read (RD) and the write (WR) which takes data and addresses inputs and two control signals called read and write. When a read command is requested,the read command go high for a short while, and we present an address to it, then that location will be read out and placed on the data bus, similarly.

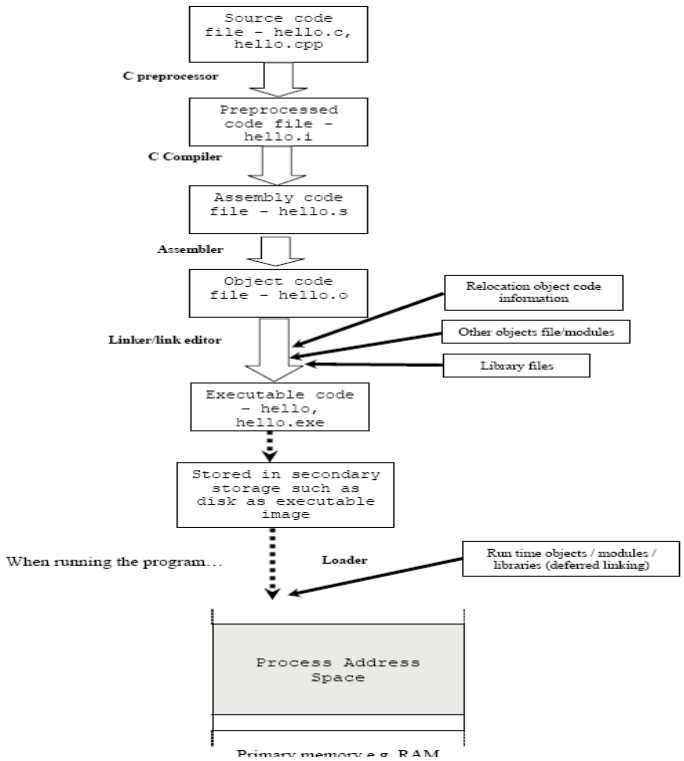

Fig: **C program building process**

COMPILER, ASSEMBLER, LINKER AND LOADER

C program construction involves four stages and develops different tools such as a preprocessor, compiler, assembler, and linker [6]. At the end it should have a single executable image that ready to be loaded by loader for the execution of the program. As mentioned in the above diagram.

1. **Pre-processing** is the first pass of any C compilation. It processes include-files, conditional compilation instructions and macros.
2. **Compilation** is the second pass. It takes the output of the preprocessor, and the source code, and generates assembler source code.
3. **Assembly** is the third stage of compilation. It takes the assembly source code and produces an assembly listing with offsets. The assembler output is stored in an object file.
4. **Linking** is the final stage of compilation. It takes one or more object files or libraries as input and combines them to produce a single file. In doing so, it resolves references to external symbols, assigns final addresses to procedures/functions and variables, and revises code and data to reflect new addresses.
5. **Loading** the executable image for program running.

Fetch Decode Execute

Fetch the instruction and execute the specified instruction and store the result. ALU (Arithmetic Logical Unit) does the calculation and store the result in the accumulator. IP Instruction pointer 8086 is 16bits in length and X86 architecture 286, 386 Pentiumtherefore we have EIP extend[2].

the microprocessor will know where to go and execute next instruction automaticallyLOCX -> EIP thereforeautomatically move to the next byte by the instruction EPI <- EIP + N (next byte). A B C D are the registers used in the processor; E is extended register.

32 Bit	8 Bit
EAX	AL, AH
EBX	BL, BH
ECX	CL, CH
EDX	DL, DH

Fig1: Register Structure

STACK POINTER ESP and BASE PONITER EBP are the stack register.

Source-Index ESI
Source-Index EDI

SI
DI

E	SP
E	BP

The register AH, AL, BH, BL, CH, CL, DH, DL (A,B,C & D) are the 16 bit register, it can be extended to the 32bits by E. therefore EAX , EBX, ECX, EDX (32bit registers)

How do we partition the memory into different segments.For example, we have a large memory. We want to ensure is the segment where the code is stored is different from wheremy data is stored where my stack data is stored and so on and, this is enabled by another set of registers which are known as the segment registers.

we have my segment registers which are simply going to demarcatewhat the different kinds of memories are in a microprocessor. example, is mycode; this could be my data; this could be my stack, andthis could be my extra segment.

EIP -> 0x0010. Where this location stored in external memory.

Complete address code = [ECS: EIP]
Stack = [ESS: ESP]
Data = ABCD
The data registers ABCD are associated with the data segment the EDI and ESI are associated with the data segment and extra segment respectively[3].

A combination of general-purpose registers andsome segmentregisters, we can address the entire memory to access code and data. The last registerthat is needed to complete this execution process

and is almost mandatory todo branching and looping is known as the flag register. So, the flag register is just anindication of a result in because of an ALU operation, arithmetic, or logical operationin the microprocessor. For example, if I subtract two resistors and it happens to go down tozero then the zero flag will be set. If I subtract two numbers and the result is negative,then you have a particular sign bit that is set.So, you can check the value of these bits in the flag register and make certain decisionsto branch and loop.

CS - CODE	**Segment 1**
DS - DATA	**Segment 2**
SS - STACK	
ES -EXTRA	**Segment 3**

Segment 4

The partition of the large memory into different segments. There are four segments named as: Code segment CS, Data Segment DS, Stack Segment SS, Extra Segment ES. The Branching and looping is known as the flag. Flag indication of a result because of the ALU Operation (arithmetic logical unit) which does the arithmetic calculation like + * / % and logical calculation like AND OR NOT XOR Operation. The Accumulator is placed inside the ALU, because ALU will only do the calculation and not store the result. Therefore, to store the temporary data we use Accumulator [4].

Fig2: Memory Segment

Instruction Set: In microprocessor there are4 types of instructions Data Transfer, ALU operation, Stack operation and Function Calls. Mnemonics are the assembly language instruction example: MOV, ADD, SUB, STA, LHLD etc.

MOV DEST, SRC; Source data, moves from Source to the Destination. The source data will not change.

MOV Ax, Bx; Source data, moves from Source to the Destination. The source data will not change.
MOV Ax, [Bx]; the content of the Bx points too.

C PROGRAMMING AND INLINE ASSEMBLY

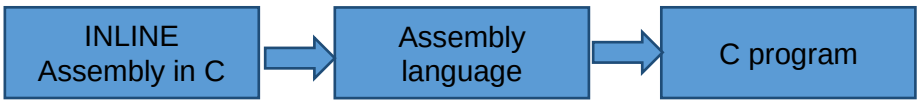

Fig3: Convert the Inline function to the C program

Inline Assembly Program convert in the assembly to the speed out. We use directive

- **- asm**
  {
  // Wrtie any assembly instruction inside the - - asm directives.

}

Example:

```
Void main ( )
{
  Int x =2;
  X = x + 4;
Printf("%d \n", x);
}
```

the Logic x = a + 2

```
- - asm {
  MOV EAX , x  // x=2
        ADD EAX, 0x0002 // 2 + 2
        MOV x, EAX     // x=4
       }
```

Variable names in the programming will remain the same in assembly language.Translate C programming to assembly language is called **INLINE**.

Int x=2, y=3, a=4, b=5;
EAx<- x * y + a – b

```
- - asm
  {
   MOV EAX, x;
  MUL Y; [EDX, EAX] <- x*y
  ADD EAX, a;
  SUB EAX, b;
  }
```

Data Types used in the C program:

Bytes of data -> CHAR
Word Data -> Short int or int
Dword of Data -> Long Int or Integer.
All the programs used in the paper are generated using x86 msvc 190 (WINE) C compiler on the complier explorer tool on the website: https://godbolt.org/. [2]

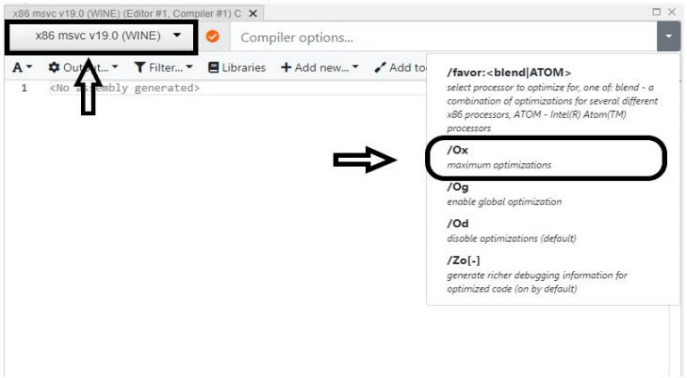

Fig4: online Tool

Inline assembly C Code

```
Int x=2, y=3, a=4, b=5;
{
    - - asm
        {
         MOV EAX, x;
        OR EAX, a;
        OR EAX, b;
        MOV EBX, x;
        AND EBX, y;
        XOR EAX, EBX;

        }
}

Void main ( )
{
Int x=3, y=4, z=0;
    - - asm {
            XOR EAX, EAX
            MOV ECX, Y
            INC ECX
            DEC ECX
            JZ LAST
        LBL: ADD EAX, X
             DEC ECX
        JNZ LBL
LAST:MOV Z, EAX
 }
Return z;
```

The expression used in the register EAX bear
EAX = (a | b| x) ⊕ (x|y)

The value for the register EAX =5;

If x=3, y= 1, a=2, b=6
The value will the register EAX = 6;

The value of the variable CZ is 12.

Conclusion:

This paper gives the understanding the integration of C and assembly language. Some of the instructions and register are being used for the explanation of the process. Programming in assembly language requires to understand the instruction set of the processor. Writing a program in machine language is like to understand the low-level details of how a machine may execute a set of instructions, fetch-execute cycle among the other instruction. Today most programmers don't deal directly with assembly language, unless the task requires direct interfacing with hardware. For example, a programmer may consider using an assembly language to write a device driver or optimize part of a game program. So, this paper gives the view of the how the C program is implemented in the machine learning language.

References:

1. https://www-personal.acfr.usyd.edu.au/tbailey/ctext/ctext.pdf
2. https://godbolt.org/
3. From C to Assembly – Linux Gazette, Issue 94, September 2003 - HiranRamakutty
4. Guide to Assembly Language Programming in Linux (Paperback) by Sivarama P. Dandamudi, Springer-Verlag, 2005
5. https://www.tenouk.com/Bufferoverflowc/Bufferoverflow1b.html
6. https://www.cs.cmu.edu/~guna/15-123S11/Lectures/Lecture27.pdf
7. https://www.javatpoint.com/assembly-program-in-c
8. Computer Systems: A Programmer's Perspective by Randal E. Bryant and David R. O'Hallaron Prentice Hall, 2003. [15-213 textbook]
9. Dr. AarushiKataria, Dr. Naveen Nandal and Dr. Ritika Malik, Shahnaz Husain -A Successful Indian Woman Entrepreneur, International Journal of Disaster Recovery and Business ContinuityVol.11, No. 2, (2020), pp. 88–93
10. Kumar, S. (2020). *Relevance of Buddhist Philosophy in Modern Management Theory*. *Psychology and Education*, Vol. 58, no.2, pp. 2104–2111.
11. Roy, V., Shukla, P. K., Gupta, A. K., Goel, V., Shukla, P. K., & Shukla, S. (2021). Taxonomy on EEG Artifacts Removal Methods, Issues, and Healthcare Applications. Journal of Organizational and End User Computing (JOEUC), 33(1), 19-46. http://doi.org/10.4018/JOEUC.2021010102
12. Shukla Prashant Kumar, Sandhu Jasminder Kaur, Ahirwar Anamika, Ghai Deepika, MaheshwaryPriti, Shukla Piyush Kumar (2021). Multiobjective Genetic Algorithm and Convolutional Neural Network Based COVID-19 Identification in Chest X-Ray Images, Mathematical Problems in Engineering, vol. 2021, Article ID 7804540, 9 pages. https://doi.org/10.1155/2021/7804540

.