

A Project Report  
On  
**Analog To Digital Time Conversion**  
For The Course  
**“Software Development Project-I”**

By  
**Meharin Choudhury**

**ID-23040**

Supervised by

**Dr. Ziaur Rahman**

**Associate Professor,**

**Department of ICT, MBSTU.**



**DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY**

**MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY**

**SANTOSH, TANGAIL-1902, Dhaka, Bangladesh**

## Declaration

This is to certify that the work presented in this project is carried out by the candidate under the supervision of **Dr. Ziaur Rahman** in the department of Information and Communication Technology, MBSTU, Tangail, Bangladesh. It is also declared that neither of this project has been submitted anywhere else for any degree or diploma.

Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Signature of Supervisor

**Dr. Ziaur Rahman**

Associate Professor

Dept. of ICT, MBSTU

## Acknowledgements

I must sense grateful to the Almighty Allah to complete the dissertation. At the outset, I would like to express gratitude to my supervisor **Dr. Ziaur Rahman**, Associate Professor, Dept. of Information and Communication Technology, MBSTU who has supported our plan to continue Software Development Project-I. I also like to express gratitude to our supervisor, for his valuable guidance and insight, encouragement, support and reliance throughout the project. However, it is not possible to acknowledge properly the effort of our honorable teacher in writing words. We are, as always, indebted to our family. The love and support of our parents remain bedrock of our life.

## Contents

Page No.

Cover Page-----	01
Declaration-----	02
Acknowledgements-----	03
 <b>Chapter -1</b> -----	 05
Introduction of project	
<b>Chapter – 2</b> -----	06-11
2.1 – Programming Language	
-- Types of programming language	
2.2 – C++ Programming Language	
-- Features of C++ Language	
-- Advantage	
-- Disadvantage	
2.3 – IDE details	
<b>Chapter – 3</b> -----	11
3.1 –Header File	
<b>Chapter – 4</b> -----	12-14
4.1 – Source Code	
<b>Chapter – 5</b> -----	15-16
5.1 – Output console	
<b>Chapter – 6</b> -----	16-17
6.1 – Conclusion	
6.2 – Future Work	

## **Chapter – 1**

### **Introduction of Project**

In this project, we will be implementing Analog time to Digital Time Conversion using the C++ programming language.

The Conversion will be Analog Time to Digital Time.

The source code is not so long, over 68 lines. It is compiled in Code::Blocks IDE with GCC compiler.

To help you understand this project better, there are lots of comments within the source code. This project is aimed to show you

## Chapter- 2

### 2.1 - Programming Language

**Programming Language :** As we know, to communicate with a person, we need a specific language, similarly to communicate with computers, programmers also need a language is called Programming language.

Before learning the programming language, let's understand what is language?

#### **What is Language?**

Language is a mode of communication that is used to share ideas, opinions with each other. For example, if we want to teach someone, we need a language that is understandable by both communicators.

#### **What is a Programming Language?**

A programming language is a computer language that is used by programmers (developers) to communicate with computers. It is a set of instructions written in any specific language ( C, C++, Java, Python) to perform a specific task.

**A programming language is mainly used to develop desktop applications, websites, and mobile applications.**

#### **Types of programming language:**

##### **1. Low-level programming language:**

Low-level language is machine-dependent (0s and 1s) programming language. The processor runs low-level programs directly without the need of a compiler or interpreter, so the programs written in low-level language can be run very fast.

Low-level language is further divided into two parts -

##### **i. Machine Language:**

Machine language is a type of low-level programming language. It is also called as machine code or object code. Machine language is easier to read because it is normally displayed in binary or hexadecimal form (base 16) form. It does not require a translator to convert the programs because computers directly understand the machine language programs.

The advantage of machine language is that it helps the programmer to execute the programs faster than the high-level programming language.

## **ii. Assembly Language:**

Assembly language (ASM) is also a type of low-level programming language that is designed for specific processors. It represents the set of instructions in a symbolic and human-understandable form. It uses an assembler to convert the assembly language to machine language.

The advantage of assembly language is that it requires less memory and less execution time to execute a program.

## **2. High-level programming language:**

High-level programming language (HLL) is designed for developing user-friendly software programs and websites. This programming language requires a compiler or interpreter to translate the program into machine language (execute the program).

The main advantage of a high-level language is that it is easy to read, write, and maintain.

High-level programming language includes Python, Java, JavaScript, PHP, C#, C++, Objective C, Cobol, Perl, Pascal, LISP, FORTRAN, and Swift programming language.

## **A high-level language is further divided into three parts -**

### **i. Procedural Oriented programming language:**

Procedural Oriented Programming (POP) language is derived from structured programming and based upon the procedure call concept. It divides a program into small procedures called routines or functions.

Procedural Oriented programming language is used by a software programmer to create a program that can be accomplished by using a programming editor like IDE, Adobe Dreamweaver, or Microsoft Visual Studio.

The advantage of POP language is that it helps programmers to easily track the program flow and code can be reused in different parts of the program.

**Example:** C, FORTRAN, Basic, Pascal, etc.

### **03.Middle-level programming language:**

Middle-level programming language lies between the low-level programming language and high-level programming language. It is also known as the intermediate programming language and pseudo-language.

A middle-level programming language's advantages are that it supports the features of high-level programming, it is a user-friendly language, and closely related to machine language and human language.

**Example:** C, C++, language

## **2.2 - C ++ Programming Language**

C++ is a programming language that is an extension of an earlier language, C. For the most part, we will use the C subset of C++ in this course because it provides the tools that we need to explore physical data structures. A few of the language features that we will use are part of C++ but not of C.





### **Most Important Features of C Language:**

- Simple.
- Abstract Data types.
- Machine Independent or Portable.
- Mid-level programming language.
- Structured programming language.
- Rich Library.
- Memory Management.
- Quicker Compilation.

### **Advantages:**

- Object-Oriented. C++ is an object-oriented programming language which means that the main focus is on objects and manipulations around these objects. ...
- Speed. ...
- Compiled. ...
- Rich Library Support. ...

- Pointer Support. ...
- Closer to Hardware.

### **Disadvantages:**

- Object-orientated programming languages have several security issues which means that programs written in C++ aren't as safe as others.
- The pointers that are used in C++ take up a lot of memory which is not always suitable for some devices.
- Cannot support built-in code threads.

## **2.3 IDE details**

### **Code::Blocks:**

Code::Blocks is a free, open-source cross-platform IDE that supports multiple compilers including GCC, Clang and Visual C++. It is developed in C++ using wxWidgets as the GUI toolkit. Using a plugin architecture, its capabilities and features are defined by the provided plugins. Currently, Code::Blocks is oriented towards C, C++, and Fortran. It has a custom build system and optional Make support.

### **Features**

### **Compilers**

Code::Blocks supports multiple compilers, including GCC, MinGW, Digital Mars, Microsoft Visual C++, Borland C++, LLVM Clang, Watcom, LCC and the Intel C++ compiler. Although the IDE was designed for the C++ language, there is some support for other languages, including Fortran and D. A plug-in system is included to support other programming languages.

### **Code editor**

The IDE features syntax highlighting and code folding (through its Scintilla editor component), C++ code completion, class browser, a hex editor and many other utilities. Opened files are organized into tabs. The code editor supports font and font size selection and personalized syntax highlighting colors.

### **Debugger**

The Code::Blocks debugger has full breakpoint support. It also allows the user to debug their program by having access to the local function symbol and argument display, user-defined watches, call stack, disassembly, custom memory dump, thread switching, CPU registers and GNU Debugger Interface.

## **Chapter – 3**

### **3.1 - Code Header file:**

```
#include<bits/stdc++.h>
```

#### **#include<bits/stdc++.h>:**

h is a header file that we use in our code to include all the standard libraries. It is quite helpful in programming contests and situations where you want to save the time of including different header files. Using the minimum includes is a good idea from the programmer's perspective.

### 4.1 – Source Code

#### Source Code:

```
#include<bits/stdc++.h>

using namespace std;

int main()
{
    int h,m,s;
    string str;
    cout<<"Enter Your Analog Time(hh mm ss AM/PM): ";
    cin>>h>>m>>s>>str;
    if(h>12 || h<1 || m>59 || s>59)
    {cout<<"Invalid Value Entered\n"<<endl;}
    else
    {
        if(str=="AM")
        {
            if(h==12)
            {h=0;}
        }
        else if(str=="PM")
        {
```

```

        if(h!=12)
        {h=h+12;}
    }
    else
    {cout<<"Invalid Information Entered\n"<<endl;}
    cout<<"Your Converted Digital Time Is: ";
    if(h<10)
    {
        cout<<"0"<<h;
        if(m<10)
        {
            cout<<":0"<<m;
            if(s<10)
            {cout<<":0"<<s<<endl;}
            else
            {cout<<":"<<s<<endl;}
        }
    }
    else
    {
        cout<<":"<<m;
        if(s<10)
        {cout<<":0"<<s<<endl;}
        else
        {cout<<":"<<s<<endl;}
    }
}

```

```

    }
}
else
{
    cout<<h;
    if(m<10)
    {
        cout<<"0"<<m;
        if(s<10)
        {cout<<"0"<<s<<endl;}
        else
        {cout<<":"<<s<<endl;}
    }
    else
    {
        cout<<":"<<m;
        if(s<10)
        {cout<<"0"<<s<<endl;}
        else
        {cout<<":"<<s<<endl;}
    }
}
}
return 0;

```

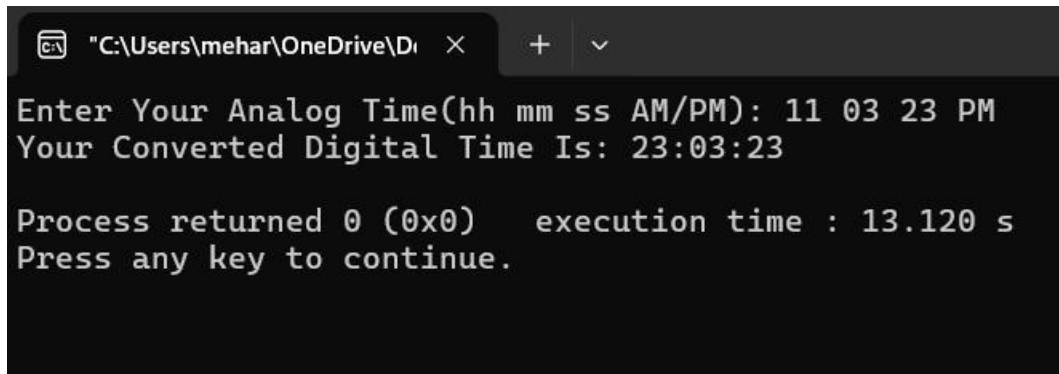
}

## Chapter – 5

### 5.1 - Output

#### Output:

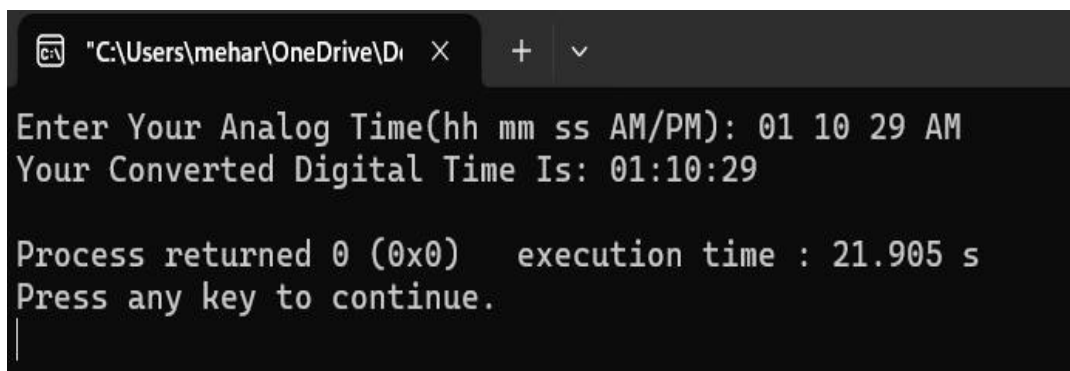
1. If we input analog time 11 03 23 PM, we get converted digital time 23:03:23 as output.



```
"C:\Users\mehar\OneDrive\Di  X + v
Enter Your Analog Time(hh mm ss AM/PM): 11 03 23 PM
Your Converted Digital Time Is: 23:03:23

Process returned 0 (0x0)   execution time : 13.120 s
Press any key to continue.
```

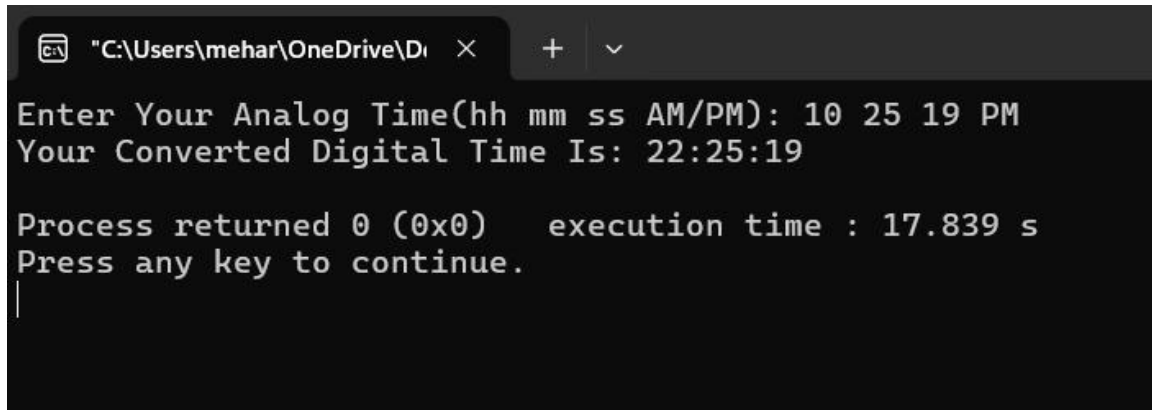
2. If we input analog time 01 10 29 AM, we get converted digital time 01:10:29 as output.



```
"C:\Users\mehar\OneDrive\Di  X + v
Enter Your Analog Time(hh mm ss AM/PM): 01 10 29 AM
Your Converted Digital Time Is: 01:10:29

Process returned 0 (0x0)   execution time : 21.905 s
Press any key to continue.
|
```

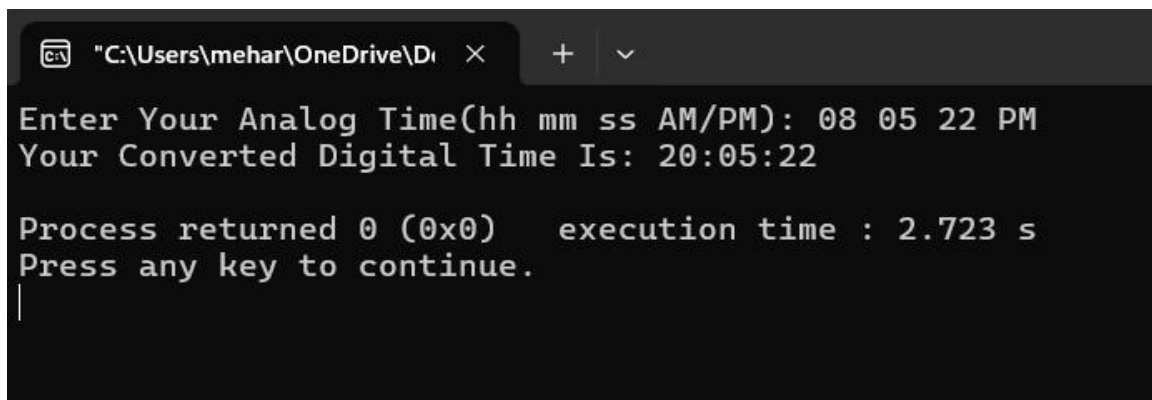
3. If we input analog time 10 25 19 PM, we get converted digital time 22:25:19 as output.



```
"C:\Users\mehar\OneDrive\Di" X + v
Enter Your Analog Time(hh mm ss AM/PM): 10 25 19 PM
Your Converted Digital Time Is: 22:25:19

Process returned 0 (0x0)    execution time : 17.839 s
Press any key to continue.
|
```

4. If we input analog time 08 05 22 PM, we get converted digital time 20:05:22 as output.



```
"C:\Users\mehar\OneDrive\Di" X + v
Enter Your Analog Time(hh mm ss AM/PM): 08 05 22 PM
Your Converted Digital Time Is: 20:05:22

Process returned 0 (0x0)    execution time : 2.723 s
Press any key to continue.
|
```

## Chapter – 6

### 6.1 – Conclusion



For developing this project we faced some difficulties which are by the directions of our honorable supervisor sir. We are still working it for adding some additional features to make this project more user friendly.

## **6.2– Future work**

**Integration with AI:** Integrating AI and machine learning algorithms into ADC systems can enhance performance through adaptive filtering and noise reduction, improving signal quality and conversion accuracy.

**Hybrid Systems:** Future designs might incorporate both analog and digital signal processing techniques, leading to hybrid converters that leverage the strengths of both approaches for better performance.

**Software-Defined Converters:** As systems become more software-driven, the concept of software-defined analog-to-digital conversion could emerge, allowing for flexible adaptation to varying signal types and conditions.