Benchmarks are quite important in PLC’s CPU design, that laydown guidelines for the processor architecture design, the ability to measure and make tradeoffs in micro-architectural decisions. This paper presents benchmarking methodology for an industrial controller by developing and running standard application programs on controllers or runtime environments in the light of IEC-61131-3 programming standard in order to assess the controller’s efficiency pertaining to Pakistan’s industrial process control and automation needs.

Keywords—benchmarking; industrial automation; IEC 61131; PLC; PAC

I. INTRODUCTION

An industrial control system is considered as the brain of any petrochemical, oil and gas, food processing industry. In the past there were never any reliable guidelines available to end user of a control system to determine which system will fulfill its requirements. Therefore, more or less unavailability of standard tests procedures to fulfill the needs of consumer to compare the performance of the industrial PLCs and PACs. The aforementioned requirement of standard benchmarking is the baseline for our research work. This research will establish benchmarks; highlights cost effectiveness and comparative performance statements between two widely used world brands of PLCs & PACs. PACs are the technology of future and will eventually replace the PLCs. The power of the PAC system is its capabilities of embedded industrial grade central processing unit and open protocol architecture.

In control world computation, benchmarking is a process of running a ladder program (IEC-61131-3), structure text program (IEC-61131-3) or a function block programs (IEC-61131-3) on an industrial controller to grade its relative performance on similar grounds and basis for a specific application. The term ‘benchmark’ in most of the cases was interpreted as the process of running a program on hardware to compare apples with apples. However, the task here in industrial control becomes handy when the programmer has to deal with a variety of coding tools and operating environments referred as integrated development environment to code and compile program. Benchmarking methodology developed here is supposed to laydown a foundation to evaluate the performance of an industrial programmable logic controller as the application required by industry user varies drastically from industry to industry and still evolving.

Benchmarks took an important place in industrial controllers CPU design that allowed the manufacturer of a CPU to make tradeoffs in order to deliver a best value proposition for the industrial end user and better value for money. For instance, a manufacturer of an industrial controller by utilizing the benchmark may be able to pinpoint the algorithms that are critical for an application. Benchmarks execution; while developing controllers give the clues to manufacturers by utilizing a simulator to mimic the application program.

Manufacturers of the industrial controllers usually refer to those standards and benchmarks which appreciate their efficiencies. Considering the critical nature of the application it is highly recommended to use the same workload as intended. For the last two decades PLCs are proved to be standard controllers to process safety critical applications in petrochemical plants. Verification procedures of logic processing power derived from computer sciences [1] don’t fulfill the dynamically changing and complex situations due to the following three reasons mainly; ease of access, supremacy of continuous dynamics and complexity.

In traditional benchmarking process the program is usually evaluated in the context of time required to execute a code of one thousand lines. Which does not depict the real time consumed to perform a program in relation to a specific application. Hence, the similar approach is no more valid for an industrial controller efficiency evaluation [2].

The “benchmarking” will be used to develop a methodology to analyze the performance of famous industrial PACs & PLCs. The users of industrial automation devices will now be able to analyze the performance of their industrial controllers.

The benchmarking methodology chiefly will cover the following goals of programming:

1) Performance analysis of a particular platform and its application [3].

2) Performance evaluation based on objectives but on a variety of industrial programmable controller [3].

II. CHALLENGES

The benchmarking involves an iterative process of running benchmarks on industrial controllers to assess the relative performance of one controller to another. There are a number of challenges associated with the benchmarks and a few of those challenges are:

---

Abstract — Benchmarks are quite important in PLC’s CPU design, that laydown guidelines for the processor architecture design, the ability to measure and make tradeoffs in micro-architectural decisions. This paper presents benchmarking methodology for an industrial controller by developing and running standard application programs on controllers or runtime environments in the light of IEC-61131-3 programming standard in order to assess the controller’s efficiency pertaining to Pakistan’s industrial process control and automation needs.
1) Considering industrial benchmarks available to manufacturers they may tweak with parameters to boost their controller's performance.

2) Speed is considered as the key parameter to evaluate the efficiency of industrial controller. However, there are other factors that are ignored for instance:

i. Reliability & availability: may be defined in terms of MTBF (Mean Time Between Failure).

ii. Security: sometimes controllers are connected with intranet to communicate the production parameters and intranet may be connected to internet. An intruder may make our life worst if we do not have any firewall between control and external world.

iii. Execution integrity: can be integrated with other devices easily?

iv. Serviceability: may be defined in terms of MTTR (Mean Time To Repair).

v. Scalability: can be scaled up based on the expanding need of customer.

vi. Trade-offs are there to bridge the quality and service required for applications.

vii. TCO (Total cost of ownership) has been not properly addressed by benchmarks [4]

viii. Electrical power consumption of industrial controllers.

3) It is not possible with the benchmarks to run multiple applications programs at the same time.

4) Perception of industrial controller's may be different then the benchmark is suggesting.

5) It has been proved with the empirical results that most the controllers performance degraded drastically at a utilization of more than seventy percent and it is recommended not to utilize more than thirty percent if user is considering a small expansion or modification project in near future and not to utilize more than fifty to sixty percent for efficient response.

6) The prime challenge is to effectively use the standards the specification and implementation analysis of PLCs programs [2]. PLC are generally considered as industrial computers utilized to automate the manufacturing industry such as, petrochemical, oil and gas, food and pharmaceutical, etc.

III. IEC 61131

IEC 61131 standards were time-honored to carry all the encoding languages being used, directives set and dissimilar concepts to be had in the turf of control system and automation world at a consistent level. The tremendous assortment of PLC concepts has led to inaptness between the several PLC platforms and vendors. The evident outcome was tremendous investment in training on hardware, virtual systems and software.

Standard IEC 61131-3 led to the standardization of various encoding languages and laid the road as a set of principles, the communication among programmable controllers, the directives set, encoding system, the management and organization of the projects. The principal benefit by means of engaging IEC 61131 is the vendor’s compliance to match controllers and encoding systems, is capable to suit all the available platforms and the deployment of same constructs. This ultimately leads to reduction in automation system TCO.

The major modifications that have arrived with IEC 61131-3 are:

1) Proclamation of variables is now structured and aligned with the regular variables proclamation in higher end encoding languages.

2) Proclamation of user defined data types is permissible under this coding standard.

3) Global versus local data variables has its own scope definition.

4) Symbolic representation means encoding in this standard.

A. Configuration, Resources and Tasks

To have a better understanding, let's have a look (see Figure 1) at the software model, as defined in the standard

![Fig 1. Software Model of IEC 61131-3 Standard](image)

1) Configuration - Configuration level allows solving a meticulous control dilemma in hand at one time that can be devised as a configuration. Configuration is explicit to a meticulous type of control system and may vary vendor to vendor that includes an array of the hardware, i.e. dealing out with system resourcefulliness, system capacities and addresses of memory pertaining to I/Os channels. Within a configuration, perhaps may define one or more resources.

2) Resources - Resources may be defined as the dispensation capability to carry out an IEC program. A resource may constitute one or more tasks in it.

3) Tasks - Tasks control the sequential or event based programs or function blocks execution. Tasks execute a routine associated with an event based on the change in a variable or on the other hand can be executed periodically.

4) Programs - IEC 61131-3 defined languages with a predefined syntax which is supposed to be followed by programmers and constitutes different elements. In general program represents a chronological order of network of functions and function blocks that have the capability to exchange data. The vital building blocks are functions and
function blocks which consists of a data structure and an algorithm.

A traditional controller consists of a resource, operating one task, controlling a program and it might be executing a closed loop. Openness has been added to the conventional controller’s structure after the incorporation of IEC 61131-3 for future.

B. Program Organization Units

POUs major parts are Function Blocks and Functions.

1) Programs

Relying on the above knowledge, a program can be recognized as the combination of function blocks and functions. Any encoding language can be used defined under the light of the IEC standards to build the desired program.

C. Sequential Function Chart, SFC

SFC as shown in Figure 2 is designed for those programmers who find it difficult to understand the structured text or even ladder program language. It represents a graphical chronological order of sequences. Petri Nets [3] is mother of this language and IEC 848 Grafcet is also its step mother. It includes all the documents support required to convert the logic from standard documentation set of directives to execution elements. The whole idea of the SFC is to divide the program load of control job into manageable chunks.

![Sequential Function Chart](image)

Fig 2. Sequential Function Chart Execution Model

IV. METHODOLOGY

After a thorough and detail review of IEC-61131-3 literature [2] the resulted methodology for this research is laid on the following milestones:

1) Scan Time Measurement
2) Development of Standard Routines
3) Selection of Runtime Environments
4) Time Stamping & Utilization of Inbuilt Counters
5) Application Oriented Programs
6) Data Types Oriented Programs
7) How to Test Other Programs with this Benchmark Routines

A. Scan Time Measurement

Scan time measurement of a chronological or sequential control program initially seems a very undemanding job. Time stamping can be accomplished by logging the start and end times. Common acuity is that in an application can be easily measured by tapping start and end of the operation. However, this acuity is hardly true due to the fact that the processor operations are being time sliced to perform the other operations for instance I/O processing, etc. The time measurement taken in this way would be highly inaccurate as the measurement is based on the single program. In a similar way, in control world most of the time overhead scan time is usually associated with the overall efficiency of the controller. Therefore, to address this issue there is a need to measure the scan time when there is no program or function block is being called at that specific time.

B. Development of Standard Routines

Since the method adopted for this research work is very close to PLCopen organization Technical Committee paper publication concept of benchmarking [5]. Therefore, it was mandatory to develop standard routines that can be executed on the selected runtime environments which will give us an estimate or a glance view that how a controller will respond to a specific application oriented code. The only difference is that the proposed standard by PLCopen might be biased with controller’s manufacturers [5]. The development of routines will guide the purchaser or end user with an idea of the controller’s strengths and weaknesses in line with their specific application. The mere purpose of this research is to lay down a methodology for selection of controllers according to their application rather than comparing controllers from one vendor to another. It is possible that one controller may work perfectly under a specific condition due their tradeoff on other varying conditions to capture that specific requirement or sector pertaining to application. The other area of development is related to data types such as Boolean, Integers, Double integers, floating types. There, are chances that one controller may produce efficient results on a certain data type but not on the other data types. Outcome of this part will guide the programmer to tune the codes to data types that a controller can efficiently handle.

C. Selection of Runtime Environments

Another taunting challenge for this research was the selection of the runtime environments that can truly mimic the real time controllers to understand and evaluate the performance of a controller. Two world’s most widely used runtime environments were selected from world renowned VENDORS

1) Siemens Energy & Automation, Inc
2) Rockwell Automation - Allen Bradley

The standard routines based on data types as well as specific application oriented programs can be written in standard software used for writing an application program for an industrial project. These programs run in real industry controllers in a similar manner as in a runtime environment usually known as simulators.

i. Criteria for selection of runtime environment

The first reason for the choice of runtime environment was brought in based on the cost associated with real controllers. This research was not funded by any company or benchmarking
organization. Therefore, investment in developing and testing the benchmark would be huge and is out of range of researcher, scope of this research. The second reason for choosing runtime environments was to provide the guideline for assessing the controller's efficiency and behavior in runtime environments before procuring the real controller for a specific project and to avoid the wrong selection. The third reason for selection of aforementioned runtime environments was the install base of VENDORS. Siemens controllers have wide install base in Europe whereas Allen Bradley has a wide install base in North America.

D. Application Oriented Benchmarks

Application oriented benchmarks means specific to a process of a plant or a control loop in the control process. These programs can be distributable over Digital I/O processing, Sequential processing similar to Petri nets as aforementioned, Motion control processing that is related to axes of motor motion, Data processing that is related to storage of data and feedback loop control application processing which is related to PID control one of the major function is to detect deviation and react on it as per pre decided control parameters. However, for simplicity, to avoid unnecessary complexity and considering the market understanding, the application oriented programs split into the two types of application benchmarks viz. digital I/O processing and Analog I/O Processing.

E. Time stamping & utilization of inbuilt counters

System in built counters are used to control the number of iterations up to 10,000 (ten thousands) times and system inbuilt commands are used to retrieve the internal system data in micro seconds resolution.

The following types of time stamping are being used:

1) Max Scan Time (micro seconds)
2) Last or Min Scan Time (micro seconds)

F. How to test other programs with this benchmark routines

The basic benchmark is developed by keeping in view the varying demands of loads. However, due non uniformity of syntax among various programming languages supplied by VENDOR to code the controllers integrated development environment it is not possible to build a single standard routine for all the users. Therefore, separate standard routines have been developed for each VENDOR'S system under test.

V. DEVELOPMENT OF STANDARD APPLICATION PROGRAMS

After selection of two world's renown brands runtime environments, the most critical tasks was to develop the standard routines that can be executed as benchmarks on these runtime environment controllers. The succeeding section will describe about the runtime environment and the development of standard routines. The following standard routines were considered:

1) Processing of Digital I/Os
2) Processing of Analog I/Os
3) Processing of Variable Analog and Fix Digital No.s of I/Os
4) Processing of Variable Digital and Fix Analog No.s of I/Os

A. RSLogix 5000™ – Rockwell Automation (Allen Bradley)

Rockwell Automation is a North American leading Programmable Logic/Automation Controller brand [6]. A brief review of its introduction will lead us to understand the definition of tags, organization and data type of tags, scope, and etc.

B. RS Logix Emulate 5000™ Runtime Environment - Emulator

RSLogix5000™ Emulator from SoftLogix5800 controller's family was selected. In this runtime environment we have configured required CPU and RSLinx for communication with I/O modules. The most important point here to remember is that yours emulator configuration shall be in a similar way as the program configuration in RSLogix5000™ configuration and programming environment.

1. Introduction to RSLogix Emulate 5000™

The below Figure 3 presents a snapshot of emulator to give you a feel that how it looks alike after configuring as per our requirements specific to a project.

C. Simatic S7 Manager™ - Siemens

Simatic S7 is a European leading Programmable Logic/Automation Controller brand [7].

Fig. 3. RSLogix 5000™ Emulator

Fig. 4. S7-PLCSIM for Simatic Step7™ Simulation
D. S7-PLCSim (Simatic S7™ Runtime Environment – Simulator)

In reference to manual S7-PLCSIM [8] see Figure 4 enables us to run and test our STEP 7 program on a simulated programmable logic controller and also known as the runtime environment. The simulation executes on our computer or programming device. Because the simulation exists completely within the STEP 7 software, we do not need to be connected to any S7 hardware (CPU or I/O modules). We can use S7-PLCSIM to simulate STEP 7 programs that were developed for S7-300, S7-400, and WinAC controllers.

S7-PLCSIM provides a simple interface to the STEP 7 program for monitoring and modifying different objects such as input and output variables. We can also use the various applications of the STEP 7 software while we are running our program on the simulated PLC. This allows us to use such tools as the variable table (VAT) to monitor and modify variables [8].

VI. RESULT OF BENCHMARKING STUDIES

This section explains the empirical results between the two selected vendor’s controllers efficiencies against each standard routine execution.

A. Digital I/Os Processing’s in terms of AND Operations

One of the standard routines is based on the processing of digital I/Os in AND operations. Digital I/Os are selectable in the form of steps as shown on the vertical axis of the chart with incremental steps. Initially the resolution of selection steps are kept at twenty five (25) I/Os and after one step it is being increased to fifty (50) I/Os, later steps are covered with an increment of one hundred (100) I/Os see Figure 5.

These operations are being executed 10,000 times on a controller’s runtime environment with varying loads to assess the response time which depicted charts here as under.

B. Analog I/Os Processing’s

The second standard routines are based on the processing of analog I/Os. Analog I/Os are selectable in the form of steps as shown on the vertical axis of the chart with incremental steps. Initially the resolution of selection steps is kept at fifty (50) I/Os, later steps are covered with an increment of one hundred (100) I/Os see Figure 6.

These operations are being executed 10,000 times on a controller’s runtime environment with varying loads to assess the response time which depicted charts here as under.

C. Variable Analog and Fix Digital No. s of I/O Processing’s

The third type of standard routines are based on the processing of analog I/Os while keeping digital number of I/Os in fixed proportion. In the first step fifty (50) digital I/Os kept fix and analog I/Os are selectable in the form of steps as shown on the vertical axis of the chart with incremental steps. Initially the resolution of selection steps was kept at fifty (50) I/Os, later steps are covered with an increment of one hundred (100) I/Os see Figure 7. These operations are being executed 10,000 times on a controller's runtime environment with varying loads to assess the response time which depicted charts here as under.
D. Variable Digital and Fix Analog No.s of I/Os Processing’s

The fourth type of standard routines are based on the processing of digital I/Os while keeping analog number of I/Os in fixed proportion. In the first step fifty (50) were kept fix and digital I/Os are selectable in the form of steps as shown on the vertical axis of the chart with incremental steps. Initially the resolution of selection steps is kept at fifty (50) I/Os, later steps are covered with an increment of one hundred (100) I/Os see Figure 8. These operations are being executed 10,000 times on a controllers runtime environments with varying loads to assess the response time which depicted charts here as under.

![Empirical Comparison of Variable Digital and Fix Analog No.s of I/Os Processing](image)

Fig. 8. Empirical Comparison of Variable Digital and Fix Analog No.s of I/Os Processing (Bar Chart)

E. Other KPIs (Key Performance Indicators) [9]

1. The logic control includes the performance/processing speed. The scalability of the system and the system wide diagnostics. The availability of multiple programming languages, reusability of code etc.

2. Other performance factors include the motion control, the flexible architecture of the system, its process control.

F. Comparison of results with state-of-art research work

Before moving towards any conclusion, we would like to draw reader's attention that the sole purpose of this research is to lay down the guideline and present an approach to cost effective automation controllers' selection for industry rather than any evaluation of vendors. This research may leave some crucial information at the end. However, a deep and thorough work effort was done in concluding this research. The end user may use this information to build its own standard routine to get the idea how fit the controller is for its application. Referring to paper by PLC Open organization [5], which refer to many other results that can also be considered for further detail work. However, for this work we have considered four scenarios suited to our research work.

VII. CONCLUSIONS

This work lays out the basic outline of the test procedure for comparing various PLC and PAC, performance, whose research can be further, expanded to benefits the industry. The intended end user may develop its own programs based on this methodology and can benefit from it by the selection of right controller for the right application. Benchmarks; highlight cost effectiveness and performance comparative statements between the widely used PLC/PAC in industry with the help of case studies.

Of course, after having quotation from various vendors the analysis of cost can be done at that point of time in hand with specific problem. It is highly recommended to utilize this methodology for end user benefits in relation to relevant application rather than pointing out inefficiencies of different vendors Because the time scan measurement is not only the criteria that should alone be focused but there other KPIs as mentioned in Section VI as performance has to be traded off to get higher edge on other iceberg.

Although, this paper covers almost all of the items that are required to benchmark the controllers but complex application like PID controls, cascade controls and communication overhead in network can be a good point to start further research in the same domain.

REFERENCES