Journal of Advances in Electronic and Electric Engineering

Volume No. 12

Issue No. 1

January - April 2024

ENRICHED PUBLICATIONS PVT. LTD

S-9, IInd FLOOR, MLU POCKET,
MANISH ABHINAV PLAZA-II, ABOVE FEDERAL BANK,
PLOT NO-5, SECTOR-5, DWARKA, NEW DELHI, INDIA-110075,
PHONE: - + (91)-(11)-47026006

Journal of Advances in Electronic and Electric Engineering

Aims and Scope

Advances in Electronic and Electric Engineering is a Journal that publishes original research papers in the fields of electronics and electrical engineering and the recent advances in these fields. It is open to all researchers from all kind of universities and organizations across the globe and aimed at the increasing important area of electronic and electrical engineering. The Journal policy is to publish high quality original scientific articles and reviews with permission from the Editorial Board. Papers must be written in English. They must not have been previously published and should not be under consideration for publication elsewhere. The principal aim of the journal is to bring together the latest research and development in various fields of science and technology such as electrical engineering, electrotechnics, electric machines modeling and design, control of electric drive systems, non-conventional energy conversion, sensors, electronics, communications, data transmission, energy converters, transducers modeling and design, electro-physics, nanotechnology, computer science, artificial intelligence, pattern recognition, knowledge engineering, process control theory and applications, distributed systems, computer networks and software engineering

Journal of Advances in Electronic and Electric Engineering

Journal of Advances in Electronic and Electric Engineering

(Volume No. 12, Issue No. 1, January - April 2024)

Contents

Pipelined Implementation Of Aes – 128 Encryption Module On Reconfigurable Logic

[1]Lokesh Namdeo, [2]Prof. Himanshu Nautiyal, [3]Prof. Sangeeta Shukla
[1,2,3]Department of Electronics & Communication Engineering, Sagar Institute of Research & Technology, Bhopal

A B S T R A C T

This paper presents the hardware implementation of 128 bit AES encryption. The implementation is optimized in order to reduce delay. In order to reduce delay pipelined architecture is employed. Also sequential shifters in shift_row process are replaced by combinational shifters to increase maximum operating frequency. The target device is Virtex-5 XC5VLX50-2FF676 speed grade -2

Keywords—AES – 128, FPGA, Rijndael Algorithm, FIPS – 197, Pipelined architecture

I. INTRODUCTION

With the development of information technology and widespread used, the security of sensitive data on Internet is especially important. Traditional cryptographic methods have failed to meet the requirements, especially to its security, speed and efficiency. The advanced encryption standard (AES) was adopted to replace the data encryption standard (DES) in 2000. AES specifies a federal information processing standard (FIPS) approved cryptographic algorithm that can be used to protect electronic data. AES is an unclassified publicly disclosed encryption algorithm available royalty free worldwide. This standard specifies the Rijndael algorithm. It is a symmetric block cipher that can encrypt and decrypt information. The data block that AES encrypt/decrypt is of 128 bit using 128, 192 or 256 bit cipher key. The original Rijndael algorithm support variable data block size and cipher block size but it was not taken by AES. So the Rijndael algorithm is taken as 128, 192 and 256 by AES and hence it is called AES 128, AES 192 and AES 256.

AES can be implemented in software or hardware but, hardware implementation is used in real time application. Main goal of AES hardware implementation is to minimize hardware and lower the power consumption also maintain high throughput at highest operating frequency.

AES hardware implementation is very reliable, fast and conveniently suitable for high speed applications. It does not require system resources used in software during encryption or decryption process. Economically AES hardware implementation has low costs compared to software implementation which requires update. Hardware encrypted drives can easily reset which reduces down time in erasing data which gives better system performance. [1]

II. AES FRAMEWORK

Table 1 shows the structure of Rijndael Algorithm adopted by AES. AES uses the data block of 128 bits and Cipher key of 128, 192 or 256. The number of rounds for AES 128, AES 192 and AES 256 are 10, 12 and 14 respectively. In each round a same set of operations are performed [2].

Table 1: Structure of AES

AES type	Structure of AES		
	Cipher Key Length	Data Block Size	Number of rounds
AES 128	128	128	10
AES 192	192	128	12
AES 256	256	128	14

A. Encryption in AES

The process of encryption begins with the conversion of 128 bit data to a 4 x 4 state matrix of 16 bytes. Similarly the input cipher key is also converted to a 4 x 4 matrix of 16 bytes. For AES – 128, the cipher key matrix size of 16 byte is same the cipher key size 128 bits (16 bytes). For AES 192 and AES 256, the first 128 bits (16 bytes) are used in first round and the remaining bits are used in next round. The set of operations performed in each round are listed below.

1. Add_Round_key: in this operation the state matrix is xored with the cipher key matrix and a new state matrix is formed.
2. Sub_bytes: in this operation each byte of state matrix is replaced by a byte form a 256 byte table called SBOX.
3. Shift_rows: State matrix has 4 rows, in this operation the first row is not shifted, the second row is shifted left cyclically by 1 byte, the third row is shifted left cyclically by 2 bytes and the fourth row is left shifted cyclically by 3 bytes.Mix_column: A linear transformation is used in this process. The mix column is process is used in 4 columns.

In this work, AES 128 encryption module is implemented, for AES 128, the number of rounds are 10. Figure 1 shows the process of encryption for AES 128. First the input matrix is added with the cipher key. Then in round 1 to round 9, 4 operations are repeated – sub_byte, shift_rows, mix column and add_round_key. In round 10 only sub_byte, shift_rows and add_round_key operation is performed. In each add_round_key operation a new cipher key is needed. This new cipher key is generated in parallel with the encryption process using key expansion logic. The same algorithm can be used for AES 192 and AES 256, just by increasing the number of rounds to 12 and 14 respectively from 10. The key expansion logic for AES 256 is slightly different. [6]

B. Decryption in AES

Figure 2 shoes the process of decryption in AES 128.The decryption process in AES is opposite to encryption process. The set of operation needed for decryption are listed below:

1. Add_round_key: this operation is exactly same to the add_round_key operation used in encryption. The encrypted data is xored with the cipher key.
2. Inv_Sub_bytes: in this operation each byte of state matrix is replaced by a byte form a 256 byte table called inv_SBOX. This SBOX is different from the one used in encryption.
3. Inv_Shift_rows: State matrix has 4 rows, in this operation the first row is not shifted, the second row is shifted right cyclically by 1 byte, the third row is shifted right cyclically by 2 bytes and the fourth row is right shifted cyclically by 3 bytes.
4. Inv_Mix_column: A linear transformation is used in this process. The inv_mix_columns is process is used in 4 columns. This is different from the one used in encryption.

The process of decryption is slightly different from the encryption; here the input is encrypted data. First the cipher key is expanded, this input cipher key is the same input key used in encryption process, the input cipher key is expanded to form 10 new keys K1 to K10 using a process called key_schedule. The process of decryption stars from the bottom, here the K10 key is used first, then K9, K8 and so on. At last the input cipher key is used to generate the decrypted data (plain text). A total of 11 keys are used in the process of encryption and decryption. The decryption is discussed only for the sake of completion; it is not implemented in this work.

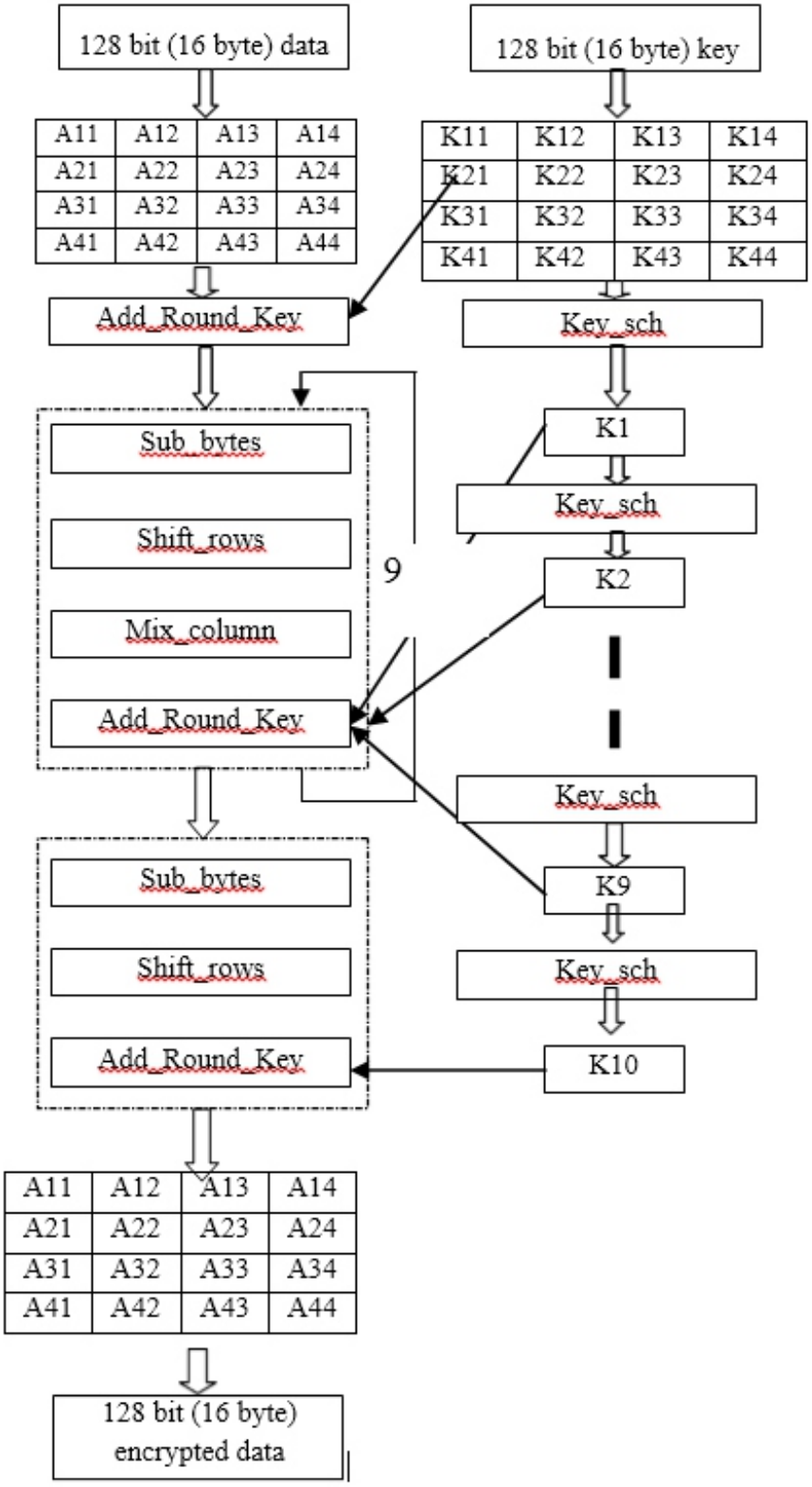


Figure 1: AES 128 encryption algorithm

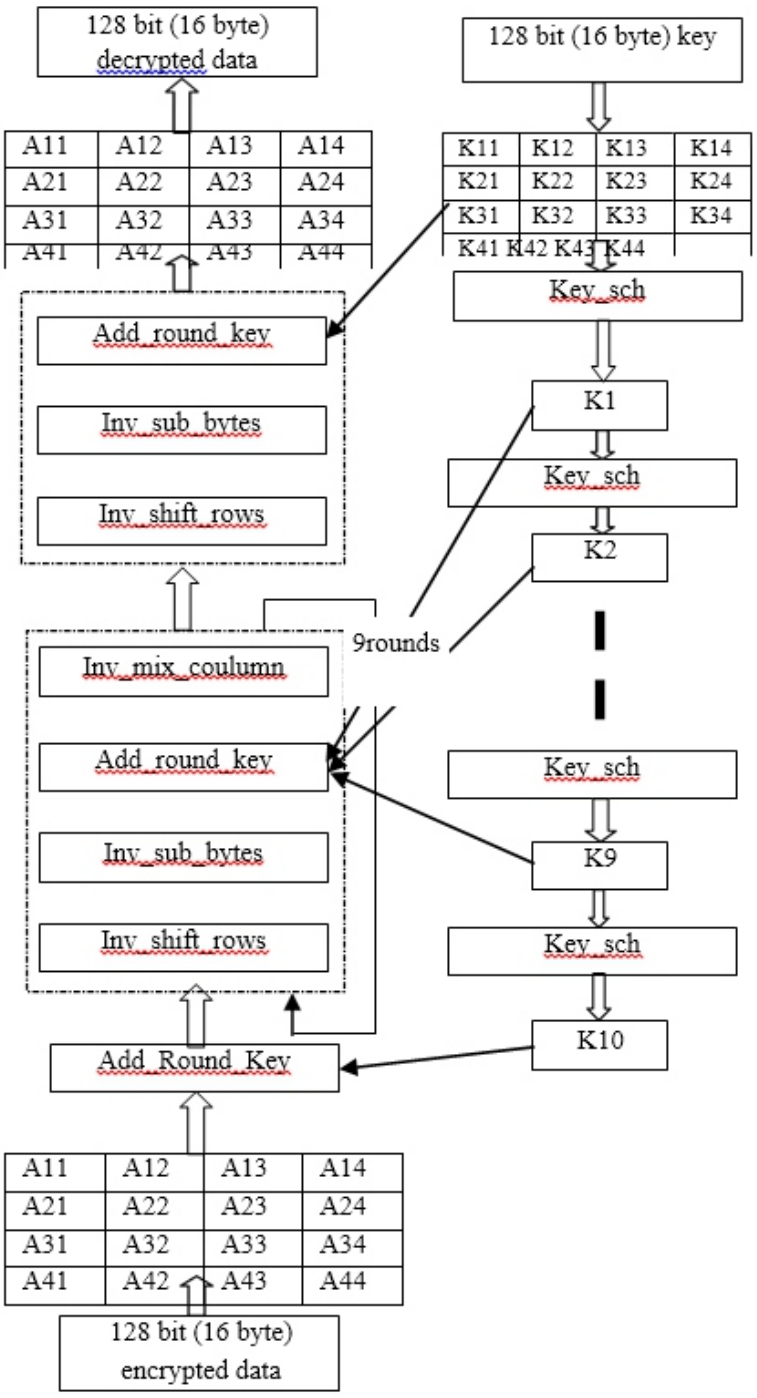


Figure 2: AES 128 decryption algorithm

III. AES – 128 ENCRYPTION MODULE PIPELINED IMPLEMENTATION

In this section the pipelined hardware implementation of AES – 128 Encryption is discussed. The high level block diagram is shown in figure 3.

AES round 1 to round 9 has four processes, 1. Sub_Bytes, 2. Shift_Rows, 3. Mix_Column and 4.Add_round_Key. AES round 10 has only sub_bytes, Shift_rows and Add_round_key process runningin it. Each of these processes are discussed in following sections. The decryption process is not implemented in this work.

1.Add_round_key unit: This unit has an array of XOR gates, which performs bit by bit XOR operation. This operation is used in Add_round unit (both encryption and decryption), AES_encryption unit (round 1 to 9), AES_encryption unit (round 10), AES_decryption unit (round 1 to 9) and AES_decryption unit (round 10).

2. AES_encryption unit (round 1 to 9): This unit has four internal units: add_round unit, Sub_bytes unit, Shift_rows unit, Mix_Column unit. add_round unit has already been discussed, remaining three units are discussed.

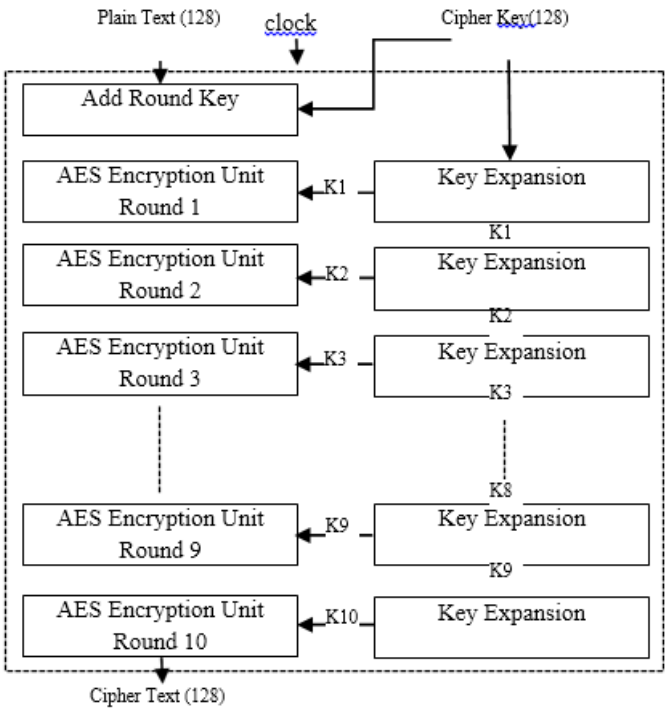


Figure 3: High Level Block Diagram AES-128-Encryption Module – Pipelined Architecture

A. Sub_byte unit: In this unit a memory table called SBOX is used. This SBOX table has 256 entries arranged in a matrix form of size 16 x 16. [6]. The input state matrix to the sub_byte unit has 16 (4 x 4) bytes, these 16 bytes are replaced by the values stored in SBOX. To replace all these 16 bytes simultaneously 16 such SBOX are used. This will reduce delay.

B. Shift_rows unit: In this unit three shifters are used to shift last three rows. The first shifter is set to shift by left by one byte, the second shifter is set to shift row by 2 bytes and the third shifter is set to shift the row by 3 bytes. Employing three parallel shifters will reduce delay. Figure 4 shows the arrangement of state matrix after shift rows.

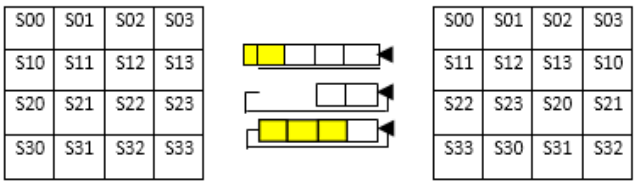


Figure 4: Arrangement of state after shift rows

C. Mix_column unit: In mix column a linear transformation is applied to the state matrix. The input state matrix to the mix_unit and the output matrix are depicted in figure 5.

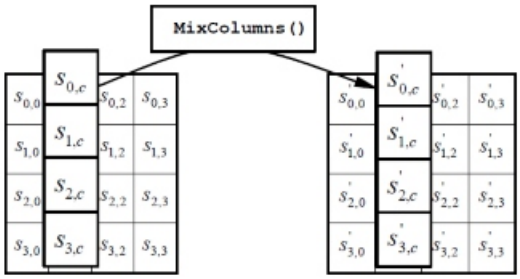


Figure 5: Arrangement of state after Mix_column

Each column of the state matrix is multiplied (galios multiplication) by constant 4 x 4 matrix and a new column matrix is formed, the new column replaces the old column.

s0c' 02 03 01 01 s0c

s1c' 01 02 03 01 s1c

s2c' 01 01 02 03 s2c

s3c' 03 01 01 02 s3c

S0c' = ({02}.Soc) xor ({03}.S1c) xor S2c xor s3c

S0c' = ({02}.Soc) xor (S1c xor ({02}.S1c)) xor S2c xor s3c

The original equation requires x_time(2) and x_time(3) operation, whereas in this work we have replaced all x_time(3) operation by a x_time(2) and XOR gate. X_time(3) is comparatively complex to x_time(2), this results in the reduction of FPGA resources.

The remaining three equations are listed below along with their reduced form.

$$S1c' = s0c \text{ xor } (\{02\}.\ S1c) \text{ xor } (\{03\}.S2c) \text{ xor } s3c$$

$$S1c' = S0c \text{ xor } (\{02\}.\ S1c) \text{ xor } (\{02\}.s20) \text{ xor } s20 \text{ xor } s3c$$

$$S2c' = S0c \text{ xor } S1c \text{ xor } (\{02\}.S2c) \text{ xor } (\{03\}.s3c)$$

$$S2c' = S0c \text{ xor } S1c \text{ xor } (\{02\}.S2c) \text{ xor } (\{02\}.S3c) \text{ xor } s3c$$

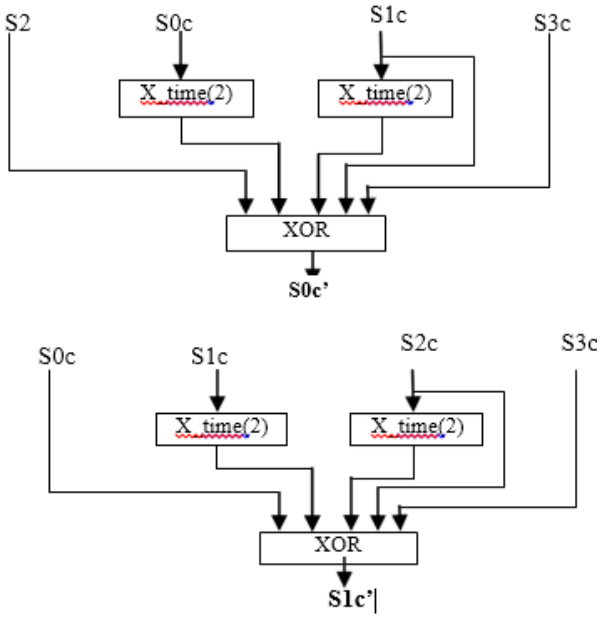
$$S3c' = (\{03\}.S0c) \text{ xor } S1c \text{ xor.} S2c \text{ xor } (\{02\}.s3c)$$

$$S3c' = (\{02\}.S0c) \text{ xor } s0c \text{ xor } S1c \text{ xor.} S2c \text{ xor } (\{02\}.s3c)$$

Figure 6 shows the logic diagram for mix_column operation for a single column.

The x_time(2) operation is simple, first the input byte is shifted by 1 bit and the MSB of input byte is checked, if it is 1 then the shifted output is XORED with 1B to produce the output else the shifted byte is the final output of x_time(2).

Figure 6 shows the mix_column operation for one single column, 4 such logic units are required to produce the mix_column output for the complete 4 x 4 matrix.



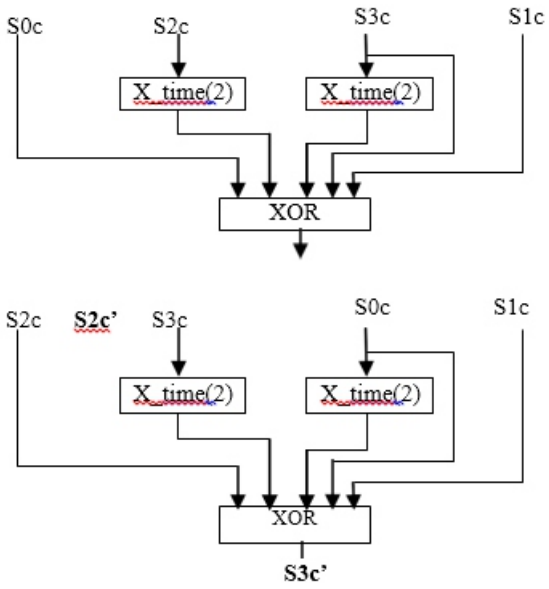


Figure 6: Mix_coulumn – Logic diagram

3. AES_encryption unit (round 10): AES last round consists of only three operation sub_bytes, Shift_rows and Add_round_key. The mix column operation is absent in this particular round.

4. Key_expansion_unit: As seen from figure 1 and figure 2 11 different cipher keys are needed in every round of encryption and decryption process. One key out of these 11 is the input cipher key and the remaining 10 keys are generated using a key expansion unit. This key expansion unit performs 4 operations rot_word, sub_word, rcon, XOR.[6]

A. Rot_word unit: The input to this operation is a row of 4 bytes. These 4 bytes are shifted left by one byte using a barrel shifter.
B. Sub_word unit: The input to this operation is also of 4 bytes, each byte of the input is replaced by a byte form the SBOX table used in encryption.
C. RCON unit: This is also a memory table which returns a 4 byte value depending upon the current round.

In this work pipelined architecture is implemented. At each clock an input is applied and corresponding output is available after 10 clock cycles. After 10 clocks the output is continuous stream. The advantage of using pipelined architecture is that the system can operate at higher data rates by introducing latency elements (registers) at each block.

IV. SIMULATION & RESULTS

In The target device used for AES – 128 encryption module implementation is XC5VLX50 - 2FF276 speed grade -2. The design is simulated using xilinx 14.1i ISIM tool. Many input vectors are used to test our design and it is found working faithfully. Figure 7 depicts the simulation result of AES-128 during encryption, output will be received after 10 clocks.

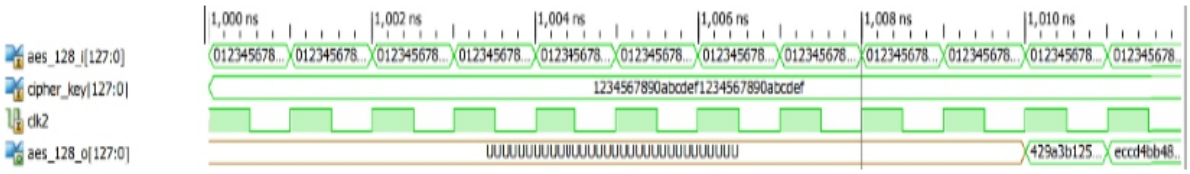


Figure 7: Simulation AES – 128 Encryption Module

The design is synthesized using xilinx XST tool. Table 2 shows the FPGA resources used, these results are compared with the previous design available in literature.

Table 2: Device Utilization Summary

Design	CLB Slices	Clock Mhz	Process
[11]	2162/	186/	Decryption/
	2057	185	Encryption
[12]	1953/	195/	Encryption/
	1947	192	Decryption
[13]	NA	60	Encryption
[14]	1005	183Mhz	Encryption
Proposed	1800	279	Encryption

In this work we have optimized the design for delay by introducing latency elements. The maximum operating frequency of this design is 278.827MHz which is much higher than the maximum operating frequency of other designs listed in table.

V. CONCLUSION

In implementation of AES – 128 is presented in this work. The target device is Virtex 5. The design is tested using XILINX 14.1 I and simulated using ISIM. Test vectors of fips document are used to check for any behavioral errors and no behavioral errors are found. We have reduced the number of x_time for the mix column operation this reduces the FPGA resource usage. Also fast shifters are used in this design, it decreases the delay. Also pipelined architecture is used to reduce the delay or in other words to increase the maximum operating frequency of the design.

In future hardware efficiency of AES – 128 can be improved by reusing resources such as using a single encryption and decryption block instead of one for round 1 to round 9 and one for round 10.

REFERENCES

[1] Hammad I, El-Sankary K, El-Masry E, "High-speed AES encryptor with efficient merging techniques," IEEE Embedded Systems Letters, 2010, pp.67-71.

[2] I ZHANG Y L, WANG X G, "Pipelined implementation of AES encryption based on FPGA," 2010 IEEE International Conference onInformation Theory and Information Security, Piscataway: IEEE,2010, pp. 170-173.

[3] FAN C-P, HWANG J-K, "Implementations of high throughput sequential and fully pipelined AES processors on FPGA." ISPACS 2007: Proceeding of 2007 International Symposium on InternationalSignal Processing and Symposium and Communication Systems, Piscataway: IEEE, 2007, pp. 353-356.

[4] SKLAVOS N, KOUFOPAVLOU O, "Architectures and VLSI implementations of the AES-proposal Rijndael," IEEE Transactions on Computers, 2002, 51(12), pp. 1454- 1459.

[5] BORKAR A M, KSHIRSAGAR R V, VYAWAHARE M V, "FPGA implementation of AES algorithm," The 3rd International Conference

[6] on Electronics Computer Technology, Piscataway: IEEE, 2011, 3, pp.401-405.

[7] Joan Daemen,Vincent Rijmen.AES Proposal:Rijdael. The RijndaelBlock Cipher.

[8] Vincent Rijmen, "Efficient implementation of the of the rijndael SBox,"2000.

[9] Fischer V, Drutarovsky M, Chodowiec P, "InvMixColunm decomposition and multilevel resource sharing in AES implementations," IEEE Transactions on Very Large Scale Integration Systems, 2005, 13(8), pp. 989-992.

[10] Chien M Ta, Chee Hong Yong, Wooi Gan Yeoh, "A 2.7mW, 0.064mm2linear-in-dB VGA with 60dB tuningrange, 100MHz bandwidth, and two DC offset cancellation loops," IEEE InternationalWorkshop on Radio Frequency Integration Technology, Austria: Graz, 2005, pp. 74-77.

[11] J.Balamurugan, Dr.E.Logashanmugam "High Speed Low Cost Implementation of Advanced Encryption Standard on FPGA" ICCET 2014.

[12] International Journal of Electronics & Telecommunication and Instrumentation Engineering, "High Speed Low Cost Implementation Of Advanced Encryption Standard On FPGA" March 2010.

[13] Bing Ji, Liejun Wang and Qinghun Yang "New version of AES-ECC Encryption based on FPGA in WSN" Journal of software engineering, 2015.

[14] Swierczynski, Pawel, et al. "Protecting against Cryptographic Trojans in FPGAs." Field-Programmable Custom Computing Machines (FCCM), 2015 IEEE 23rd Annual International Symposium on. IEEE, 2015.

Extract Transform Load Data With Etl Tools Like 'Informatica'

[1]Preeti, [2]Neetu Sharma

[1,]M. Tech., Computer Science Engineering, Ganga Institute of Technology and Management, Bahadurgarh-Jhajjar Road, Kablana, Distt. Jhajjar, Haryana

[2]HOD C.S.E Deptt., Ganga Institute of Technology and Management, Bahadurgarh-Jhajjar Road, Kablana, Distt. Jhajjar, Haryana

A B S T R A C T

As we all know business intelligence (BI) is considered to have an extraordinary impact on businesses. Research activity has grown in the last years [10]. A significant part of BI systems is a well performing Implementation of the Extract, Transform, and Load (ETL) process. In typical BI projects, implementing the ETL process can be the task with the greatest effort. Here, set of generic Meta model constructs with a palette of regularly used ETL activities, is specialized, which are called templates.

I. INTRODUCTION

We all want to load our data warehouse regularly so that it can assist its purpose of facilitating business analysis [1]. To do this, data from one or more operational systems desires to be extracted and copied into the warehouse. The process of extracting data from source systems and carrying it into the data warehouse is commonly called ETL, which stands for extraction, transformation, and loading. It is an essential phenomenon in a data warehouse. Whenever DML (data manipulation language) operations such as INSERT, UPDATE OR DELETE are issued on the source database, data extraction occurs. After data extraction and transformation have taken place, data are loaded into the data warehouse.

Extraction: The first part of an ETL process is to extract the data from the home systems. Most data warehousing projects amalgamate data from unlike source systems. Each separate system may also use a different data association format. Common data source formats are relational databases and flat files, but may contain non-relational database structures such as IMS or other data structures .Extraction converts the data into a format for transformation processing. The quantity of data is reduced by omitting any non-relevant data sets. Extraction must not negatively affect the performance of productive systems. It runs as a background task or is executed at times of low activity (e.g. during the night).

Transformation: Any transformation desirable to provide data that can be interpreted in business terms is done in the second step. Data sets are cleaned with regard to their data quality. Eventually, they are

converted to the scheme of the target database and consolidated. The transform stage applies a series of rules or functions to the extracted data to derive the data to be loaded. Some data sources will require very slight manipulation of data. Data transformations are often the most difficult and, in terms of processing time, the most costly part of the ETL process. They can range from simple data conversions to extremely complex data scrubbing techniques.

Loading: Now, the real loading of data into the data warehouse has to be done. The early Load which generally is not time-critical is great from the Incremental Load. Whereas the first phase affected productive systems, loading can have an giant effect on the data warehouse. This especially has to be taken into consideration with regard to the complex task of updating currently stored data sets. In general, incremental loading is a critical task. ETL processes can either be run in batch mode or real time. Batch jobs typically are run periodically. If intervals become as short as hours or even minutes only, these processes are called near real time. The load phase loads the data into the data warehouse. Depending on the requirements of the organization, this process ranges widely. Some data warehouses merely overwrite old information with new data.

More complex systems can maintain a history and audit trail of all changes to the data. Designing and maintaining the ETL process is often considered one of the most difficult and resource-intensive portions of a data warehouse project. Many data warehousing projects use ETL tools to manage this process. Data warehouse builders create their own ETL tools and processes, either inside or outside the database.

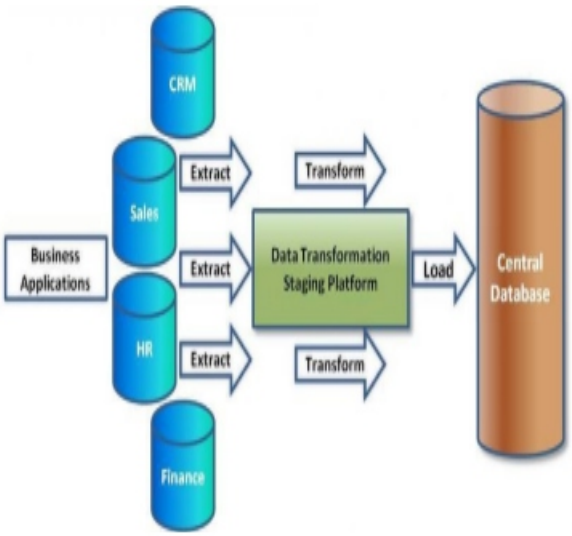


Fig. The ETL Process

ETL tools have been around for some time, have evolved, matured, and now present us with productive environments for Big Data, Data Warehouse, Business Intelligence, and analytics processing. However these are few problems with them .Such tools are very expensive and does not support small size businesses. Configuration of such tools takes lot of time. These are many ETL tools availble in the market .Some of the ETL Tools are:

- DataStage from Ascential Software
- SAS System from SAS Institute
- Informatica
- Data Integrator From BO
- Oracle Express
- Abinito
- Decision Stream From Cognos
- MS-DTS from Microsoft
- Pentaho Kettle

Informatica is the best ETL tool in the marketplace [3]. It can extract data from numerous heterogeneous sources, transforming them as per business needs and loading to target tables. It's used in Data migration and loading projects. It is a visual interface and you will be dragging and dropping with the mouse in the Designer(client Application).This graphical approach to communicate with all major databases and can move/transform data between them. It can move huge bulk of data in a very effective way. Informatica is a tool, supporting all the steps of Extraction, Transformation and Load process. Now a days Informatica is also being used as an Integration tool.

Informatica is an easy to use tool. It has got a simple visual interface like forms in visual basic. You just need to drag and drop different objects (known as transformations) and design process flow for Data extraction transformation and load. These process flow diagrams are known as mappings. Once a mapping is made, it can be scheduled to run as and when required. In the background Informatica server takes care of fetching data from source, transforming it, & loading it to the target systems/databases.

Informatica can talk with all major data sources (mainframe/RDBMS/Flat Files/XML/VSM/SAP etc), [3] can move/transform data between them. It can move huge volumes of data in a very operational way, many a times better than even bespoke programs written for specific data movement only. It can throttle the transactions (do big updates in small chunks to avoid long locking and filling the transactional log). It can effectively join data from two distinct data sources (even a xml file can be joined with a relational

table). In all, Informatica has got the ability to effectively integrate heterogeneous data sources & converting raw data into useful information.

Some facts and figures about Informatica Corporation:

- Founded in 1993, based in Redwood City, California
- 1400+ Employees; 3450 + Customers; 79 of the Fortune 100 Companies
- NASDAQ Stock Symbol: INFA; Stock Price: $18.74 (09/04/2009)
- Revenues in fiscal year 2008: $455.7M
- Headquarters: Redwood City, CA
- Offices: N. & S. America, Europe, Asia Pacific
- Government organizations in 20 countries
- Partners: Over 400 major SI, ISV,OEM and On Demand

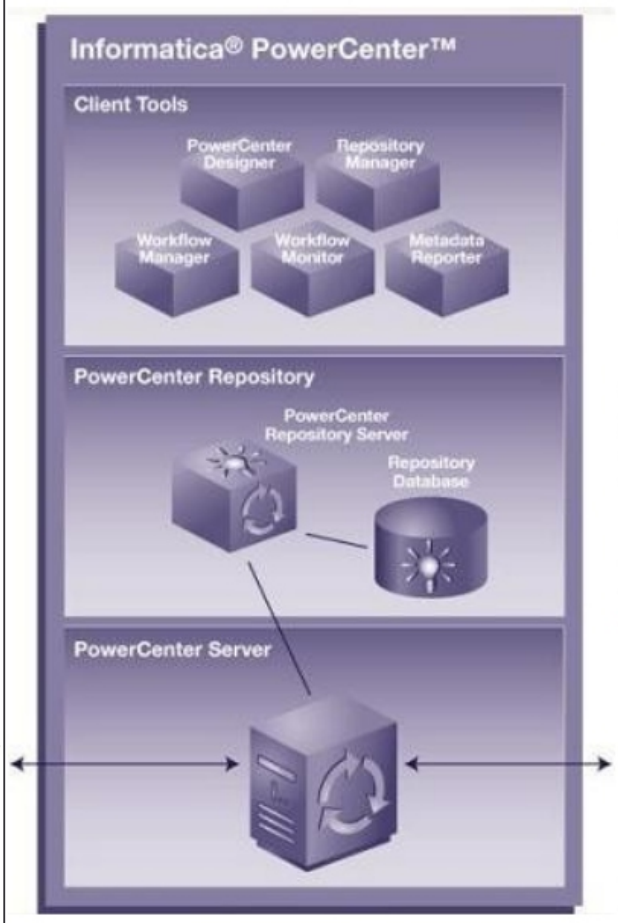
2. COMPONENTS OF INFORMATICA

Informatica provides the following integrated components:

Informatica repository. The Informatica repository is at the center of the Informatica suite. You create a set of metadata tables within the repository database that the Informatica applications and tools access. The Informatica Client and Server access the repository to save and retrieve metadata. The PowerCenter repository resides on a relational database. The repository database tables contain the instructions required to extract, transform, and load data. PowerCenter Client applications access the repository database tables through the Repository Server. You add metadata to the repository tables when you perform tasks in the PowerCenter Client application, such as creating users, analyzing sources, developing mappings or mapplets, or creating workflows. The PowerCenter Server reads metadata created in the Client application when you run a workflow. The PowerCenter Server also creates metadata, such as start and finish times of a session or session status[2].

You can develop global and local repositories to share metadata:

Global repository. The global repository is the hub of the domain. Use the global repository to store common objects that multiple developers can use through shortcuts. These objects may include operational or Application source definitions, reusable transformations, mapplets, and mappings.

Sources		Targets
Standard: RDBMS, Flat Files, XML, ODBC		**Standard:** RDBMS, Flat Files, XML, ODBC
Applications: SAP R/3, SAP BW, PeopleSoft, Siebel, JD Edwards, i2		**Applications:** SAP R/3, SAP BW, PeopleSoft, Siebel, JD Edwards, i2
EAI: MQ Series, Tibco, JMS, Web Services		**EAI:** MQ Series, Tibco, JMS, Web Services
Legacy: Mainframes (DB2, VSAM, IMS, IDMS, Adabas)AS400 (DB2, Flat File)		**Legacy:** Mainframes (DB2)AS400 (DB2)
Remote Sources		**Remote Targets**

Local repositories. A local repository is within a domain that is not the global repository. Use local repositories for development. From a local repository, you can create shortcuts to objects in shared folders in the global repository. These objects typically include source definitions, common dimensions and lookups, and enterprise standard transformations. You can also create copies of objects in non-shared folders.

Version control. A versioned repository can store multiple copies, or versions, of an object. Each version is a separate object with unique properties. PowerCenter version control features allow you to efficiently develop, test, and deploy metadata into production.

You can connect to a repository, back up, delete, or restore repositories using pmrep, a command line program.

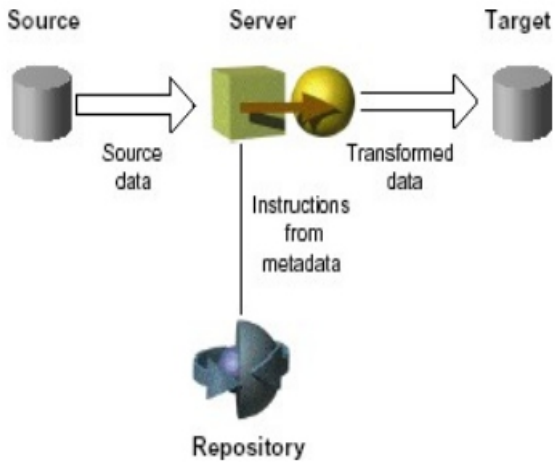
You can view much of the metadata in the Repository Manager. The Informatica Metadata Exchange (MX) provides a set of relational views that allow easy SQL access to the Informatica metadata repository.

Repository Server

The Repository Server manages repository connection requests from client applications. For each repository database registered with the Repository Server, it configures and manages a Repository Agent process. The Repository Server also monitors the status of running Repository Agents, and sends repository object notification messages to client applications. Informatica Client. Use the Informatica Client to manage users, define sources and targets, build mappings and mapplets with the transformation logic, and create sessions to run the mapping logic. The Informatica Client has three client applications: Repository Manager, Designer, and Workflow Manager.

- **Repository Server Administration Console.** Use the Repository Server Administration console to administer the Repository Servers and repositories.
- **Repository Manager.** Use the Repository Manager to administer the metadata repository. You can create repository users and groups, assign privileges and permissions, and manage folders and locks.
- **Designer.** Use the Designer to create mappings that contain transformation instructions for the **PowerCenter Server.** Before you can create mappings, you must add source and target definitions to the repository. The Designer has five tools that you use to analyze sources, design target schemas, and build source-to-target mappings:
    - **Source Analyzer.** Import or create source definitions.
    - **Warehouse Designer.** Import or create target definitions.Transformation Developer. Develop reusable transformations to use in mappings.
    - **Mapplet Designer.** Create sets of transformations to use in mappings.
    - **Mapping Designer.** Create mappings that the PowerCenter Server uses to

- **Workflow Manager.** Use the Workflow Manager to create, schedule, and run workflows. A workflow is a set of instructions that describes how and when to run tasks related to extracting, transforming, and loading data. The PowerCenter Server runs workflow tasks according to the links connecting the tasks. You can run a task by placing it in a workflow.
- **Workflow Monitor.** Use the Workflow Monitor to monitor scheduled and running workflows for each PowerCenter Server. You can choose a Gantt chart or Task view. You can also access details about those workflow runs.



InformaticaServer: The Informatica Server extracts the source data, performs the data transformation, and loads the transformed data into the targets.

The PowerCenter Server reads mapping and session information from the repository. It extracts data from the mapping sources and stores the data in memory while it applies the transformation rules that you configure in the mapping. The PowerCenter Server loads the transformed data into the mapping targets.

The PowerCenter Server can achieve high performance using symmetric multi-processing systems. The PowerCenter Server can start and run multiple workflows concurrently. It can also concurrently process partitions within a single session. When you create multiple partitions within a session, the PowerCenter Server creates multiple database connections to a single source and extracts a separate range of data for each connection, according to the properties you configure.

3. INFORMATICA PRODUCT LINE

Informatica is a powerful ETL tool from Informatica Corporation, a leading provider of enterprise data integration software and ETL softwares.

The important products provided by Informatica Corporation is provided below:

- Power Center
- Power Mart
- Power Exchange
- Power Center Connect
- Power Channel
- Metadata Exchange

- Power Analyzer
- Super Glue

Power Center & Power Mart: Power Mart is a departmental version of Informatica for building, deploying, and managing data warehouses and data marts. Power center is used for corporate enterprise data warehouse and power mart is used for departmental data warehouses like data marts. Power Center supports global repositories and networked repositories and it can be connected to several sources. Power Mart supports single repository and it can be connected to fewer sources when compared to Power Center. Power Mart can extensibily grow to an enterprise implementation and it is easy for developer productivity through a codeless environment.

Power Exchange: Informatica Power Exchange as a stand-alone service or along with Power Center, helps organizations leverage data by avoiding manual coding of data extraction programs. Power Exchange supports batch, real time and changed data capture options in main frame(DB2, VSAM, IMS etc.,), mid-range (AS400 DB2 etc.,), and for relational databases (oracle, sql server, db2 etc) and flat files in unix, linux and windows systems.

Power Center Connect: This is adding on to Informatica Power Center. It helps to extract data and metadata from ERP systems like IBM's MQSeries, Peoplesoft, SAP, Siebel etc. and other third party applications.

Power Channel: This helps to transfer large amount of encrypted and compressed data over LAN, WAN, through Firewalls, tranfer files over FTP, etc.

Meta Data Exchange: Metadata Exchange enables organizations to take advantage of the time and effort already invested in defining data structures within their IT environment when used with Power Center. For example, an organization may be using data modeling tools, such as Erwin, Embarcadero, Oracle designer, Sybase Power Designer etc for developing data models. Functional and technical team should have spent much time and effort in creating the data model's data structures(tables, columns, data types, procedures, functions, triggers etc). By using meta deta exchange, these data structures can be imported into power center to identifiy source and target mappings which leverages time and effort. There is no need for informatica developer to create these data structures once again.

Power Analyzer: Power Analyzer provides organizations with reporting facilities. PowerAnalyzer makes accessing, analyzing, and sharing enterprise data simple and easily available to decision makers.

PowerAnalyzer enables to gain insight into business processes and develop business intelligence. With PowerAnalyzer, an organization can extract, filter, format, and analyze corporate information from data stored in a data warehouse, data mart, operational data store, or other data storage models. PowerAnalyzer is best with a dimensional data warehouse in a relational database. It can also run reports on data in any table in a relational database that do not conform to the dimensional model.

Super Glue: Superglue is used for loading metadata in a centralized place from several sources. Reports can be run against this superglue to analyze meta data.

4. TYPES OF INFORMATICA PARTITIONS

Informatica provides you the option of enhancing the performance of the Informatica session by the The PowerCenter® Partitioning Option. After tuning all the performance bottlenecks we can further improve the performance by addition partitions[3].We can either go for Dynamic partitioning (number of partition passed as parameter) or Non- dynamic partition (number of partition are fixed while coding). Apart from used for optimizing the session, Informatica partition become useful in situations where we need to load huge volume of data or when we are using Informatica source which already has partitions defined, and using those partitions will allow to improve the session performance.

The partition attributes include setting the partition point, the number of partitions, and the partition types.

Partition Point: There can be one or more pipelines inside a mapping. Adding a partition point will divide this pipeline into many pipeline stages. Informatica will create one partition by default for every pipeline stage. As we increase the partition points it increases the number of threads. Informatica has mainly three types of threads –Reader, Writer and Transformation Thread. The number of partitions can be set at any partition point. We can define up to 64 partitions at any partition point in a pipeline. When you increase the number of partitions, you increase the number of processing threads, which can improve session performance. However, if you create a large number of partitions or partition points in a session that processes large amounts of data, you can overload the system.

You cannot create partition points for the following transformations:

- Source definition
- Sequence Generator
- XML Parser

- XML target
- Unconnected transformations

The partition type controls how the Integration Service distributes data among partitions at partition points. The Integration Service creates a default partition type at each partition point.
Types of partitions are:

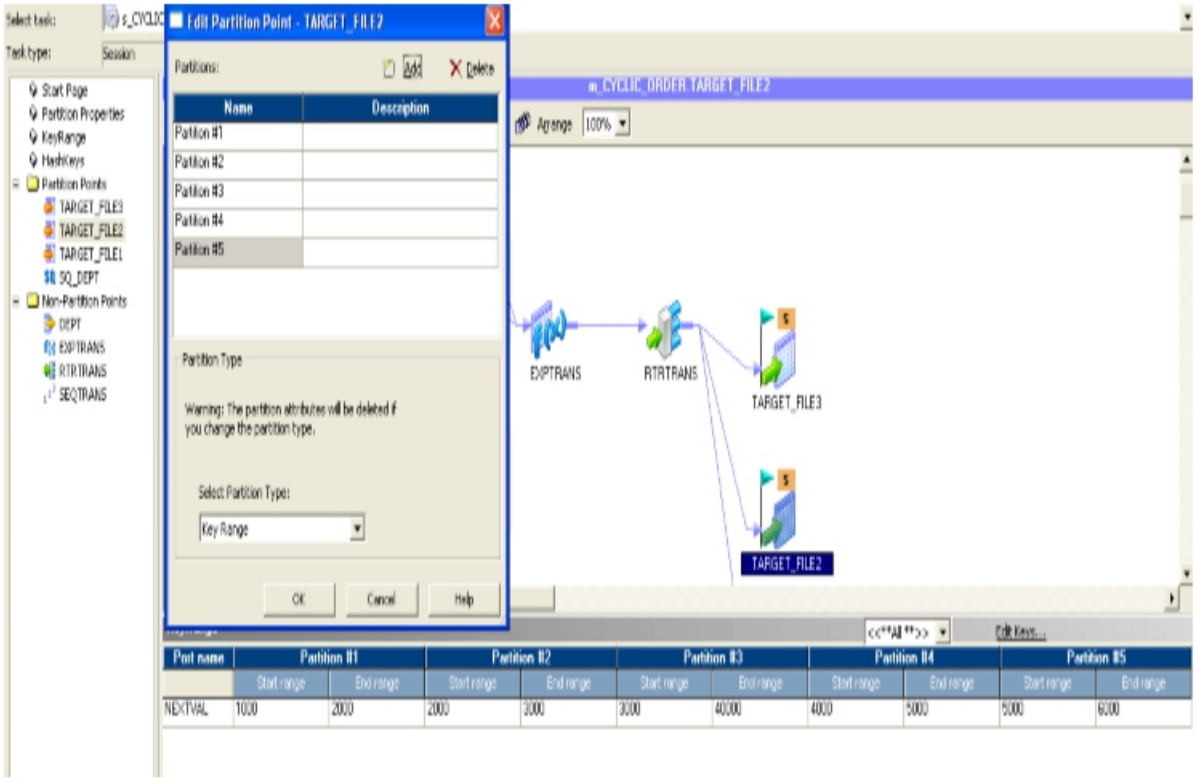
1. Database partitioning
2. Hash auto-keys
3. Hash user keys
4. Key range
5. Pass-through
6. Round-robin

Database Partitioning: For Source Database Partitioning, Informatica will check the database system for the partition information if any and fetches data from corresponding node in the database into the session partitions. When you use Target database partitioning, the Integration Service loads data into corresponding database partition nodes.

Use database partitioning for Oracle and IBM DB2 sources and IBM DB2 targets.

Pass through: Using Pass through partition will not affect the distribution of data across partitions instead it will run in single pipeline which is by default for all your sessions. The Integration Service processes data without redistributing rows among partitions. Hence all rows in a single partition stay in the partition after crossing a pass-through partition point.

Key range: Used when we want to partition the data based on upper and lower limit. The Integration Service will distribute the rows of data based on a port or set of ports that we define as the partition key. For each port, we define a range of values. Based on the range that we define the rows are send to different partitions.



Round robin partition is used to when we want to distributes rows of data evenly to all partitions Hash auto-keys: The Integration Service uses a hash function to group rows of data among partitions. The Integration Service groups the data based on a partition key.

Hash user keys: The Integration Service uses a hash function to group rows of data among partitions. We define the number of ports to generate the partition key.

5. TRANSFORMATIONS

Informatica Transformations: A transformation is a repository object that generates, modifies, or passes data. The Designer provides a set of transformations that perform specific functions. A transformation is a repository object that generates, modifies, or passes dataThe Designer provides a set of transformations that perform specific functions.Data passes into and out of transformations through ports that you connect in a mapping.

Transformations can be of two types:

Active Transformation: An active transformation can change the number of rows that pass through the transformation, change the transaction boundary, can change the row type. For example, Filter, Transaction Control and Update Strategy are active transformations. The key point is to note that

Designer does not allow you to connect multiple active transformations or an active and a passive transformation to the same downstream transformation or transformation input group because the Integration Service may not be able to concatenate the rows passed by active transformations However, Sequence Generator transformation (SGT) is an exception to this rule[4]. A SGT does not receive data. It generates unique numeric values. As a result, the Integration Service does not encounter problems concatenating rows passed by a SGT and an active transformation.

Aggregator	performs aggregate calculations
Filter	serves as a conditional filter
Router	serves as a conditional filter (more than one filters)
Joiner	allows for heterogeneous joins
Source qualifier	represents all data queried from the source

Passive Transformation: A passive transformation does not change the number of rows that pass through it, maintains the transaction boundary, and maintains the row type. The key point is to note that Designer allows you to connect multiple transformations to the same downstream transformation or transformation input group only if all transformations in the upstream branches are passive.

Expression	performs simple calculations
Lookup	looks up values and passes to other objects
Sequence generator	generates unique ID values
Stored procedure	calls a stored procedure and captures return values
Update strategy	allows for logic to insert, update, delete, or reject data

6.1 TYPES OF TRANSFORMATION

1. Expression Transformation:

You can use the Expression transformation to calculate values in a single row before you write to the target. For example, you might need to adjust employee salaries, concatenate first and last names, or convert strings to numbers. You can use the Expression transformation to perform any non-aggregate calculations. You can also use the Expression transformation to test conditional statements before you output the results to target tables or other transformations.

Calculating Values: To use the Expression transformation to calculate values for a single row, you must include the following ports:

- Input or input/output ports for each value used in the calculation. For example, when calculating the total price for an order, determined by multiplying the unit price by the quantity ordered, the input or input/output ports. One port provides the unit price and the other provides the quantity ordered.
- Output port for the expression. You enter the expression as a configuration option for the output port. The return value for the output port needs to match the return value of the expression. For information on entering expressions, see "Transformations" in the Designer Guide. Expressions use the transformation language, which includes SQL-like functions, to perform calculations.

You can enter multiple expressions in a single Expression transformation. As long as you enter only one expression for each output port, you can create any number of output ports in the transformation. In this way, you can use one Expression transformation rather than creating separate transformations for each calculation that requires the same set of data.

2. Joiner Transformation:

You can use the Joiner transformation to join source data from two related heterogeneous sources residing in different locations or file systems. Or, you can join data from the same source. The Joiner transformation joins two sources with at least one matching port. The Joiner transformation uses a condition that matches one or more pairs of ports between the two sources. If you need to join more than two sources, you can add more Joiner transformations to the mapping.The Joiner transformation requires input from two separate pipelines or two branches from one pipeline.

The Joiner transformation accepts input from most transformations. However, there are some limitations on the pipelines you connect to the Joiner transformation. You cannot use a Joiner transformation in the following situations:

- Either input pipeline contains an Update Strategy transformation.
- You connect a Sequence Generator transformation directly before the Joiner transformation

The join condition contains ports from both input sources that must match for the PowerCenter Server to join two rows. Depending on the type of join selected, the Joiner transformation either adds the row to the result set or discards the row. The Joiner produces result sets based on the join type, condition, and input data sources. Before you define a join condition, verify that the master and detail sources are set for optimal performance. During a session, the PowerCenter Server compares each row of the master

source against the detail source. The fewer unique rows in the master, the fewer iterations of the join comparison occur, which speeds the join process. To improve performance, designate the source with the smallest count of distinct values as the master. You can improve session performance by configuring the Joiner transformation to use sorted input. When you configure the Joiner transformation to use sorted data, the PowerCenter Server improves performance by minimizing disk input and output. You see the greatest performance improvement when you work with large data sets. When you use a Joiner transformation in a mapping, you must configure the mapping according to the number of pipelines and sources you intend to use. You can configure a mapping to join the following types of data:

- **Data from multiple sources.** When you want to join more than two pipelines, you must configure the mapping using multiple Joiner transformations.
- **Data from the same source.** When you want to join data from the same source, you must configure the mapping to use the same source.

Perform joins in a database when possible.

Performing a join in a database is faster than performing a join in the session. In some cases, this is not possible, such as joining tables from two different databases or flat file systems. If you want to perform a join in a database, you can use the following options:

- Create a pre-session stored procedure to join the tables in a database.
- Use the Source Qualifier transformation to perform the join.

Join sorted data when possible.

You can improve session performance by configuring the Joiner transformation to use sorted input. When you configure the Joiner transformation to use sorted data, the PowerCenter Server improves performance by minimizing disk input and output. You see the greatest performance improvement when you work with large data sets.

For an unsorted Joiner transformation, designate as the master source the source with fewer rows. For optimal performance and disk storage, designate the master source as the source with the fewer rows. During a session, the Joiner transformation compares each row of the master source against the detail source.

3. Rank Transformation:

The Rank transformation allows you to select only the top or bottom rank of data. You can use a Rank

transformation to return the largest or smallest numeric value in a port or group. You can also use a Rank transformation to return the strings at the top or the bottom of a session sort order. During the session, the PowerCenter Server caches input data until it can perform the rank calculations. You connect all ports representing the same row set to the transformation. Only the rows that fall within that rank, based on some measure you set when you configure the transformation, pass through the Rank transformation.

You can also write expressions to transform data or perform calculations. As an active transformation, the Rank transformation might change the number of rows passed through it. You might pass 100 rows to the Rank transformation, but select to rank only the top 10 rows, which pass from the Rank transformation to another transformation.

Rank Caches

During a session, the PowerCenter Server compares an input row with rows in the data cache. If the input row out-ranks a cached row, the PowerCenter Server replaces the cached row with the input row. If you configure the Rank transformation to rank across multiple groups, the PowerCenter Server ranks incrementally for each group it finds.

Rank Transformation Properties:

- Enter a cache directory.
- Select the top or bottom rank.Select the input/output port that contains values used to determine the rank. You can select only one port to define a rank.
- Select the number of rows falling within a rank.
- Define groups for ranks, such as the 10 least expensive products for each manufacturer.

The Rank transformation changes the number of rows in two different ways. By filtering all but the rows falling within a top or bottom rank, you reduce the number of rows that pass through the transformation. By defining groups, you create one set of ranked rows for each group

4. Router Transformation:

A Router transformation is similar to a Filter transformation because both transformations allow you to use a condition to test data. A Filter transformation tests data for one condition and drops the rows of data that do not meet the condition. However, a Router transformation tests data for one or more conditions and gives you the option to route rows of data that do not meet any of the conditions to a default output group. If you need to test the same input data based on multiple conditions, use a Router transformation

in a mapping instead of creating multiple Filter transformations to perform the same task. The Router transformation is more efficient. For example, to test data based on three conditions, you only need one Router transformation instead of three filter transformations to perform this task. Likewise, when you use a Router transformation in a mapping, the PowerCenter Server processes the incoming data only once.

5. Lookup Transformation:

Use a Lookup transformation in a mapping to look up data in a flat file or a relational table, view, or synonym. You can import a lookup definition from any flat file or relational database to which both the PowerCenter Client and Server can connect[6]. You can use multiple Lookup transformations in a mapping. It compares Lookup transformation port values to lookup source column values based on the lookup condition. Pass the result of the lookup to other transformations and a target.

You can use the Lookup transformation to perform many tasks, including:

- **Get a related value.** For example, your source includes employee ID, but you want to include the employee name in your target table to make your summary data easier to read.
- **Perform a calculation.** Many normalized tables include values used in a calculation, such as gross sales per invoice or sales tax, but not the calculated value (such as net sales).
- **Update slowly changing dimension tables.** You can use a Lookup transformation to determine whether rows already exist in the target.

  You can configure the Lookup transformation to perform the following types of lookups:
- **Connected or unconnected.** Connected and unconnected transformations receive input and send output in different ways.
- **Relational or flat file lookup.** When you create a Lookup transformation, you can choose to perform a lookup on a flat file or a relational table.
- **Cached or uncached.** Sometimes you can improve session performance by caching the lookup table. If you cache the lookup, you can choose to use a dynamic or static cache. By default, the lookup cache remains static and does not change during the session. With a dynamic cache, the PowerCenter Server inserts or updates rows in the cache during the session. When you cache the target table as the lookup, you can look up values in the target and insert them if they do not exist, or update them if they do.

Note: If you use a flat file lookup, you must use a static cache.

Using Sorted Input

When you configure a flat file Lookup transformation for sorted input, the condition columns must be grouped. If the condition columns are not grouped, the PowerCenter Server cannot cache the lookup and fails the session. For best caching performance, sort the condition columns. The Lookup transformation also enables an associated ports property that you configure when you use a dynamic cache.

6. ADVANTAGES OF INFORMATICA

A comprehensive integration platform that promotes code standardization, unifies collaboration between business and IT roles, and provides capabilities that handle the highvolume and wide variety of today's business data. The Informatica Platform has eight distinct technologies designed to be a true industrial strength ETL solution. These include:

- Messaging
- Complex Event Processing
- B2B Data Exchange
- Cloud Data Integration
- Enterprise Data Integration
- Application Lifecycle Management
- Data Quality
- Master Data Management

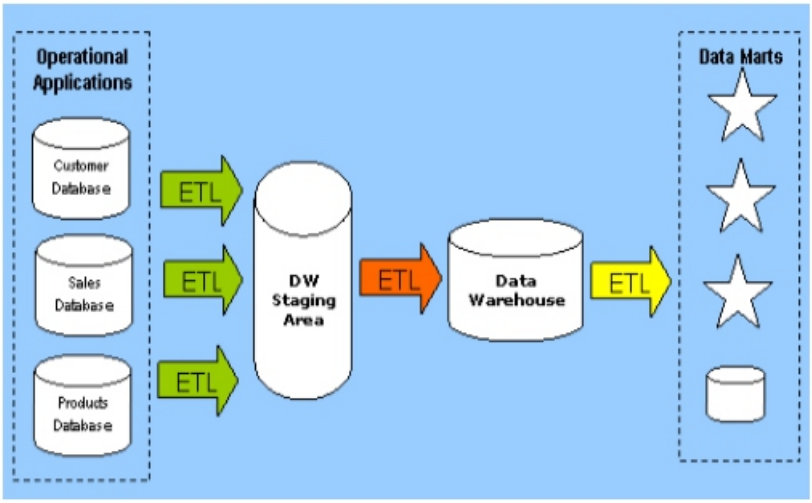
- **Improve network speed**. Slow network connections can slow session performance. Have your system administrator determine if your network runs at an optimal speed. Decrease the number of network hops between the PowerCenter Server and databases.
- **Use multiple PowerCenter Servers.** Using multiple PowerCenter Servers on separate systems might double or triple session performance.
- **Use a server grid.** Use a collection of PowerCenter Servers to distribute and process the workload of a workflow. For information on server grids.
- **Improve CPU performance.** Run the PowerCenter Server and related machines on high performance CPUs, or configure your system to use additional CPUs.Configure the PowerCenter Server for ASCII data movement mode. When all character data processed by the PowerCenter Server is 7-bit ASCII or EBCDIC, configure the PowerCenter Server for ASCII data movement mode[5].
- **Check hard disks on related machines**. Slow disk access on source and target databases, source and target file systems, as well as the PowerCenter Server and repository machines can slow session performance. Have your system administrator evaluate the hard disks on your machines.
- **Reduce paging.** When an operating system runs out of physical memory, it starts paging to disk to free physical memory. Configure the physical memory for the PowerCenter Server machine to minimize paging to disk.
- **Use processor binding.** In a multi-processor UNIX environment, the PowerCenter Server may use a large amount of system resources. Use processor binding to control processor usage by the PowerCenter Server.

7. DATA STAGING

The data staging area is the data warehouse workbench. It is the place where raw data is brought in, cleaned, combined, archived, and eventually exported to one or more data marts. The purpose of the data staging area is to get data ready for loading into a presentation server (a relational DBMS or an OLAP engine). A staging area, or landing zone, is an intermediate storage area used for data processing during the extract, transform and load (ETL) process. The data staging area sits between the data source(s) and the data target(s), which are often data warehouses, data marts, or other data repositories [1]

Data staging areas are often transient in nature, with their contents being erased prior to running an ETL process or immediately following successful completion of an ETL process. There are staging area architectures, however, which are designed to hold data for extended periods of time for archival or

troubleshooting purposes. Staging areas can be implemented in the form of tables in relational databases, text-based flat files (or XML files) stored in file systems or proprietary formatted binary files stored in file systems.[2] Staging area architectures range in complexity from a set of simple relational tables in a target database to self-contained database instances or file systems.[3] Though the source systems and target systems supported by ETL processes are often relational databases, the staging areas that sit between data sources and targets need not also be relational databases.[4]



The Data Warehouse Staging Area is temporary location where data from source systems is copied. A staging area is mainly required in a Data Warehousing Architecture for timing reasons. In short, all required data must be available before data can be integrated into the Data Warehouse.[1]

The staging area in Business Intelligence is a key concept. The role of this area is to have a secure place to store the source systems data for further transformations and cleanings. Why do we do that?
Because:

- It minimizes the impact on the source systems (you don't want to re-extract everything from the source systems if your ETL failed).
- It can be used for auditing purposes (we store the data that we process).
- It eases the development process (you don't need to be bound to the operational servers).

The data staging area of the data warehouse is both a storage area and a set of processes commonly referred to as extract-transformation-load (ETL).[2] The data staging area is everything between the operational source systems and the data presentation area. It is somewhat analogous to the kitchen of a restaurant, where raw food products are transformed into a fine meal. In the data warehouse, raw operational data is transformed into a warehouse deliverable fit for user query and consumption.

Similar to the restaurant's kitchen, the backroom data staging area is accessible only to skilled professionals. The data warehouse kitchen staff is busy preparing meals and simultaneously cannot be responding to customer inquiries. Customers aren't invited to eat in the kitchen. [3]

It certainly isn't safe for customers to wander into the kitchen. We wouldn't want our data warehouse customers to be injured by the dangerous equipment, hot surfaces, and sharp knifes they may encounter in the kitchen, so we prohibit them from accessing the staging area. Besides, things happen in the kitchen that customers just shouldn't be privy to. The key architectural requirement for the data staging area is that it is off-limits to business users and does not provide query and presentation services.

Once the data is extracted to the staging area, there are numerous potential transformations, such as cleansing the data (correcting misspellings, resolving domain conflicts, dealing with missing elements, or parsing into standard formats), combining data from multiple sources, deduplicating data, and assigning warehouse keys. These transformations are all precursors to loading the data into the data warehouse presentation area.[1] The data staging area is dominated by the simple activities of sorting and sequential processing. In many cases, the data staging area is not based on relational technology but instead may consist of a system of flat files. After you validate your data for conformance with the defined one-to-one and many-to-one business rules, it may be pointless to take the final step of building a fullblown third-normal-form physical database. However, there are cases where the data arrives at the doorstep of the data staging area in a third-normal-form relational format. In these situations, the managers of the data staging area simply may be more comfortable performingthe cleansing and transformation tasks using a set of normalized structures.

8. DATA STAGING PROCESS

A staging area, or landing zone, is an intermediate storage area used for data processing during the extract, transform and load (ETL) process. The data staging area sits between the data source(s) and the data target(s), which are often data warehouses, data marts, or other data repositories.[1]The data staging process imports data either as streams or files, transforms it, produces integrated, cleaned data and stages it for loading into data warehouses, data marts, or Operational Data Stores.[2]
First, Kimball distinguishes two data staging scenarios.

In (1) a data staging tool is available, and the data is already in a database. The data flow is set up so that it comes out of the source system, moves through the transformation engine, and into a staging database. The flow is illustrated in Figure One.

Figure One -- First Data Staging Scenario

In the second scenario, begin with a mainframe legacy system. Then extract the sought after data into a flat file, move the file to a staging server, transform its contents, and load transformed data into the staging database.

Figure Two illustrates this scenario.

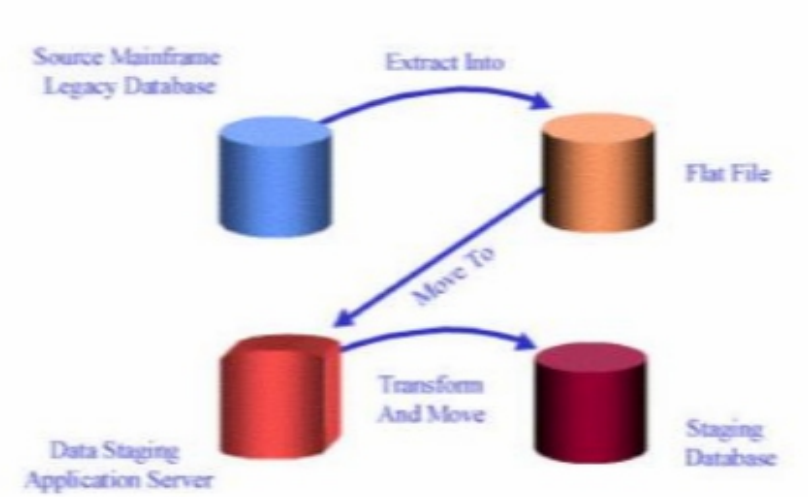


Figure Two -- Second Data Staging Scenario

We assume that the data staging area is not a query service. In other words, any database that is used for querying is assumed to be physically downstream from the data staging area. If the legacy data is already available in a relational database, then it may make sense to perform all the processing steps within the relational framework, especially if the source relational database and the eventual target presentation database are from the same vendor.[3] This makes even more sense when the source database and the target database are on the same physical machine, or when there is a convenient high-speed link between them. However, there are many variations on this theme, and in many cases it may not make sense to load the source data into a relational database. In the detailed descriptions of the processing steps, we will see that almost all the processing consists of sorting, followed by a single, sequential pass through either one or two tables.[1]

This simple processing paradigm does not need the power of a relational DBMS. In fact, in some cases, it may be a serious mistake to divert resources into loading the data into a relational database when what is needed is sequential flat-file processing. Similarly, we will see that if the raw data is not in a normalized entity-relationship (ER) format, in many cases it does not pay to load it into an ER physical model simply to check data relationships. The most important data integrity steps involving the enforcement of one-to-one and one-to-many relationships can be performed, once again, with simple sorting and sequential processing. It is acceptable to create a normalized database to support the staging processes; however, this is not the end goal. The normalized structures must be off-limits to user queries because they defeat understandability and performance. As soon as a database supports query and presentation services, it must be considered part of the data warehouse presentation area. By default, normalized databases are excluded from the presentation area, which should be strictly dimensionally structured. Regardless of whether we're working with a series of flat files or a normalized data structure in the staging area, the final step of the ETL process is the loading of data. Loading in the data warehouse environment usually takes the form of presenting the quality-assured dimensional tables to the bulk loading facilities of each data mart.[2] The target data mart must then index the newly arrived data for query performance. When each data mart has been freshly loaded, indexed, supplied with appropriate aggregates, and further quality assured, the user community is notified that the new data has been published.

The data staging area of the data warehouse is both a storage area and a set of processes commonly referred to as extract-transformation-load (ETL). The data staging area is everything between the operational source systems and the data presentation area. The process of extracting data from source systems and bringing it into the data warehouse is commonly called ETL, which stands for extraction, transformation, and loading.[3] It is a fundamental phenomenon in a data warehouse . Whenever DML (data manipulation language) operations such as INSERT, UPDATE OR DELETE are issued on the source database, data extraction occurs. After data extraction and transformation have taken place, data are loaded into the data warehouse.

Extraction: The first part of an ETL process is to extract the data from the source systems. Most data warehousing projects consolidate data from different source systems. Each separate system may also use a different data organization format. Common data source formats are relational databases and flat files, but may include non-relational database structures such as IMS or other data structures[4] .Extraction converts the data into a format for transformation processing. The amount of data is reduced by omitting any non- relevant data sets. Extraction must not negatively affect the performance of productive systems. Extracting means reading and understanding the source data and copying the data needed for the data warehouse into the staging area for further manipulation

Transformation: Any transformation needed to provide data that can be interpreted in business terms is done in the second step. Data sets are cleaned with regard to their data quality. Eventually, they are converted to the scheme of the target database and consolidated. The transform stage applies a series of rules or functions to the extracted data to derive the data to be loaded. Some data sources will require very little manipulation of data. Data transformations are often the most complex and, in terms of processing time, the most costly part of the ETL process. They can range from simple data conversions to extremely complex data scrubbing techniques.[2]

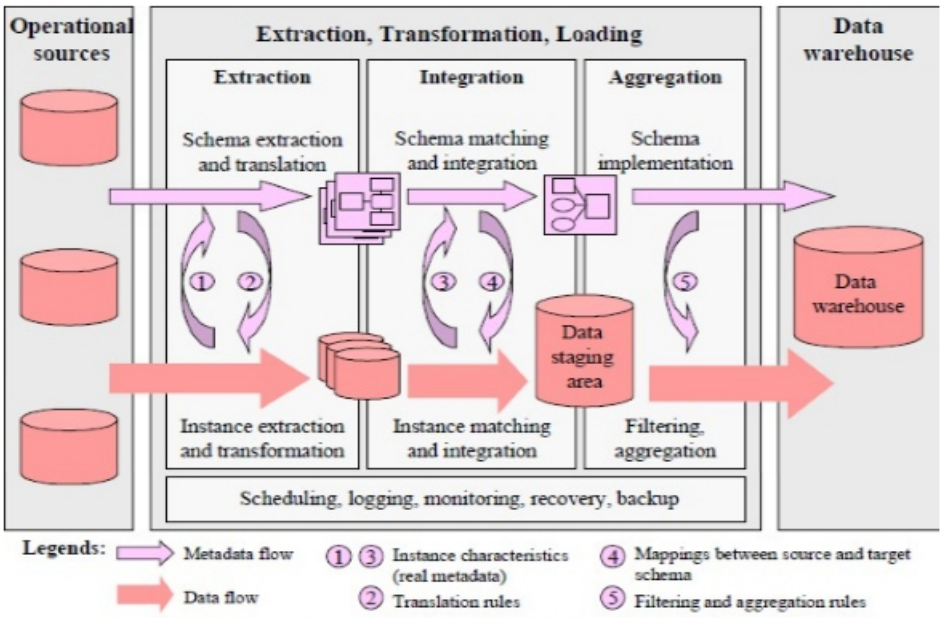


Figure 1. Steps of building a data warehouse: the ETL process

Loading: Finally, the actual loading of data into the data warehouse has to be done. The Initial Load which generally is not time-critical is distinguished from the Incremental Load. Whereas the first phase affected productive systems, loading can have an immense effect on the data warehouse. This especially has to be taken into consideration with regard to the complex task of updating currently stored data sets. In general, incremental loading is a critical task. ETL processes can either be run in batch mode or real time. Batch jobs typically are run periodically. If intervals become as short as hours or even minutes only, these processes are called near real time. The load phase loads the data into the data warehouse. Depending on the requirements of the organization, this process ranges widely. Some data warehouses merely overwrite old information with new data.[3]

Unfortunately, there is still considerable industry consternation about whether the data that supports or results from this process should be instantiated in physical normalized structures prior to loading into the presentation area for querying and reporting. These normalized structures sometimes are referred to in the industry as the enterprise data warehouse; however, we believe that this terminology is a misnomer

because the warehouse is actually much more encompassing than this set of normalized tables. The enterprise's data warehouse more accurately refers to the conglomeration of an organization's data warehouse staging and presentation areas. [4]

A normalized database for data staging storage is acceptable. However, we continue to have some reservations about this approach. The creation of both normalized structures for staging and dimensional structures for presentation means that the data is extracted, transformed, and loaded twice—once into the normalized database and then again when we load the dimensional model. Obviously, this two-step process requires more time and resources for the development effort, more time for the periodic loading or updating of data, and more capacity to store the multiple copies of the data.[1] At the bottom line, this typically translates into the need for larger development, ongoing support, and hardware platform budgets. Unfortunately, some data warehouse project teams have failed miserably because they focused all their energy and resources on constructing the normalized structures rather than allocating time to development of a presentation area that supports improved business decision making. While we believe that enterprise-wide data consistency is a fundamental goal of the data warehouse environment, there are equally effective and less costly approaches than physically creating a normalized set of tables in your staging area, if these structures don't already exist.

9. CONCLUSION

The Informatica solution for enterprise data warehousing is proven to help IT departments implement data marts and departmental data warehouses and readily scale them up to enterprise data warehousing environments. This solution serves as the foundation for all data warehousing and enterprise data warehousing projects. It accelerates their deployment, minimizing costs and risks, by ensuring that enterprise data warehouses are populated and maintained with trustworthy, actionable, and authoritative data.

Data is and has been from the beginning created, stored, and retrieved by disparate, incompatible systems. Between 30% and 35% of all the data in the industry is still on mainframes, in languages and data structures that are archaic and generally unavailable.

The wave of specialty applications—HR, sales, accounting, ERP, manufacturing—have all contributed their share to the chaos. Informatica PowerCenter is the ETL tool that empowers an IT organization to implement highly scalable, high-performance data processing maps using a graphical interface that generates proprietary intermediate code[9]. This mapping code, when coupled with a defined data workflow plan can then be scheduled for a variety of execution strategies. Widely accepted as an

industry front runner, Informatica boasts high productivity and low cost. In truth, this may be just the opposite. Perhaps high productivity at a cost is more accurate; in terms of experienced developers, administrators, and license fees. Companies who choose to use Informatica usually have very large IT teams and budgets.

REFERENCES

[1] http://dwhlaureate.blogspot.in/2012/07/informaticaetl-extract-transform-and.html

[2] http://www.dbbest.com/blog/extract-transform-load-etl-technologies-part-2/

[3] http://www.informatica.com/us/#fbid=rHNS5hx2rKv

[4] http://www.ventanaresearch.com/blog/commentblog.aspx?id=3524

[5] http://www.slideshare.net

[6] http://www.rpi.edu/datawarehouse/docs/ETL-Tool-Selection.pdf

[7] http://en.wikipedia.org/wiki/Informatica

[8] http://www.etltool.com/etl-vendors/powercenter-informatica/

[9] http://www.techopedia.com/definition/25983/informatica-powercenter

[10] www.ijsrp.org/research-paper-0214/ijsrp-p2688.pdf

Energy Minimization Techniques Over Multicore Processing System: A Review

[1]K. Nagalakshmi , [2]PN. Gomathi

[1,] Department of Computer Science and Engineering,
Hindusthan Institute of Technology, Coimbatore, Tamilnadu, India
[2]Department of Computer Science and Engineering,
Vel Tech Dr. RR & Dr. SR Technical University, Chennai, Tamilnadu, India

A B S T R A C T

Energy efficiency and processor performance have become key metrics in the designing of multicore computational systems. Due to breaking down of Moore's law, increasing energy savings without compromising raw performance is considered as a major limiting factor in multicore architecture. Recent technological advances in energy minimizing methods of multicore system substantially meet the contradictory demands of low power, low cost, small area and outstanding performance. This paper aims at ascertaining more competent energy-minimizing techniques for managing energy consumption of multicore processor through investigations. We highlight the necessity of the energy savings techniques and study several novel technologies to focus their pros and cons. This paper is intended to serve the researchers and architects of multicore processors in accumulating ideas about the energy savings techniques and to incorporate it in near future for more effective fabrications.

Keywords: Clock Gating; DVFS; Energy Efficient Design; Multicore Processor; Task Scheduling;

I. INTRODUCTION

At present, multicore architectures (MCA) are becoming dominant design paradigm which assimilates two or more processing elements (cores) in a single die for higher performance computing. Proliferation of heavy computational requirements of real time applications in MCAs leads severe energy efficiency and performance constraint to provide quality of service (QoS) to the users. Thus, designing the MCAs to resolve power-performance tradeoff is a challenging endeavor. The research and design community have invested significant efforts in exploring several energy efficient technologies to scale their performance and make sure reliability, prolonged existence and acceptance in wide range of applications.

As stated in the Moore's law, the number of transistors fabricated in a single chip approximately doubles every 18 to 24 months [1], resulting in an exponential increase in transistor density. This indicates that the speed (clock frequency) of the processor will also double in every 18 months. However we cannot enjoy this exponential growth continuously due to its increasing power density on chip which prevents all the cores to be switched on simultaneously. This utilization barrier is called as Dark silicon, is driving

the emergence of heterogeneous MCAs [2]. Multicore processors can be categorized into three types: Homogeneous [3, 6, 8], Heterogeneous [10, 11] and Dynamic reconfigurable processors [12]. Conventionally, most of the general purpose MCAs are built with identical cores. All these cores consist of same micro-architectural innovations (i.e., cache memory, out-of- order execution, speculation, pipeline, branch prediction configuration, etc.) and are able to operate under same instruction set architecture (ISA). This type of architecture is called as a homogeneous or symmetric multicore architecture (SMP). It is easy to design and implement as we just need to duplicate the core.

INTEL CORE i7 [3], AMD PHENOM [4] and SUN NIAGARA [5] multicore architectures are general purpose SMPs with large cache memories. These processors are designed for general purpose desktop and server applications where energy efficiency is not a primary concern. In contrast, the homogeneous architectures XMOS-XS1 [6] and ARM-CORTEX [7] are specially designed for mobile devices, where energy efficiency is an increasing concern. Some of the MCAs are developed for high performance computing. Therefore, they employ larger number of cores. For example, AMD RADEON 700 GPU [8] contains 160 cores while NVIDIA G200 [9] contains 240 cores. Though homogeneous cores are simple to design, easy to implement and provide regular software environments; they cannot deliver required performance and energy efficiency for different real time applications. Real time applications with different QoS encourage the computer architects and software designers to exploit architecture innovations and design heterogeneous multicore processors.

Asymmetric Multicore Processor (AMP) or Heterogeneous processor implements mixture of non-identical processing elements that are asymmetric in their underlying principles and performance. The cores are varying in size and complexity, but they are designed to cooperate with each other to increase the performance of the system. These designs provide an efficient solution for dark silicon era and also increase reliability, performanceand energy efficiency of the applications. A typical AMP will integrate small (slow) cores to process simple tasks in an energy efficient way, and complex (fast) cores to provide higher performance.

ARM big.LITTLE [10] and Cell BE [11] are renowned examples for AMP architectures. Cell BE is extensively used in gaming devices and computing platforms aiming high performance computing. ARM big.LITTLE is designed for mobile platforms where complex, performance driven, quad core Cortex A15 assembly is combined with simple, power-optimized, quad core Cortex A7 assembly to provide peak performance.

Dynamic heterogeneous multicore architectures are able to reconfigure itself at run time to regulate their performance, speed and complexity level based on application requirements. It has the ability to resolve the power-performance tradeoff by integrating efficient hardware with flexible software. By introducing adaptability and hardware flexibility, this dynamic architectures can achieve high performance within the energy budget and therefore to meet the QoS requirements of real time applications. Flexible heterogeneous Multicore processor (FMC) is an eminent example of dynamic reconfigurable architecture that can deliver both an increased throughput for uniformly distributed parallel workloads and outstanding performance for fluctuating real time tasks *12]. Depending upon the application requirements, the FMC can scale up and down its computing resources such as memory engines (ME), functional units, and pipelines that are anticipated to improved performance.

The evolution of multicore designs opens up a new space of research called energy saving techniques. Reducing the peak and average energy consumption could have a positive impact on performance of multicore processors, starting from circuit level to system level. In the last decade, several researches have been keen to explore various energy saving techniques in multicore regime. The remaining part of our survey is structured as follows: Section II presents the basics of power dissipation in multicore domain and also highlights the necessity of energy saving techniques. Section III provides an overview and taxonomy of classic power management techniques and explores some of these techniques in detail. Finally, Section IV is arranged to provide the conclusions of our investigation.

II. BACKGROUND

A. Basics of Power dissipation

Nowadays, multicore is ubiquitous both in general purpose and application specific computing systems starting from smartphones to commercial servers. Power dissipation is an important constraint because it increases the temperature and cooling costs, reduces reliability, and degrades performance. Power consumed by CMOS devices can be resolved into three parts [13] as shown in equation (1).

$$P_T = P_{st} + P_{dy} + P_{sc} \qquad (1)$$

Where P_T is total power dissipation, P_{st} is static or leakage power due to leakage of a transistor's bias currents. P_{dy} is dynamic power due to switching of transistors. P_{sc} is power dissipation related to concurrent conduction of p type and n type transistors.

$$P_{st} = V.I_L \qquad (2)$$

Where V denotes source voltage, I_L is the transistor leakage current. Usually, leakage power contributes 20 to 40 % of the total power dissipation [14]. Dynamic power is a prevalent factor as compared to other two components in equation (1). It can be denoted as follows

$$P_{dy} = \beta C_L . V^2.f \qquad (3)$$

Where β is activity factor, C_L indicates the effective load capacitance and f is the switching speed. To simplify equation (3), it is assumed that the clock speed is linearly proportional to the source voltage. If we apply this notion to the above equation (3) then, the reduction in source voltage and switching speed lowers dynamic power cubically.

$$P_{dy} \alpha V^3 \qquad (4)$$

Short circuit power is calculated by the following equation

$$P_{sc} = V.I_{sc} \qquad (5)$$

Here I_{sc} represents the short circuit current flowing from supply to ground. Short circuit power is comparatively trivial for static CMOS circuits.

B. Failure of Dennard scaling

For almost 30 years, the computing community has realized a stable performance evolution in uniprocessor, motivated by Moore's Law *1] and Classical (Dennard) scaling [15]. But now this curve is slowed down and came to halt due to memory wall, power wall and Instruction Level Parallelism (ILP) wall [16]. Under Dennard scaling, the power requirements per unit space can remain constant across semiconductor generations. According to this principle [15], with a linear dimension scaling ratio of 0.707, the transistor count could doubles (Moore's Law), frequency increases by 40%, but the power consumed per transistor is reduced by half keeping the total chip power constant in every two years [17]. From equation (3), the power density in a chip area A is measured as follows

$$\text{Dynamic Power density} = (\beta C_{Load} . V^2. f)/A \qquad (6)$$

As we move towards the next generation of IC manufacturing technology, the linear size of an IC gets scaled by 0.707. The same scaling ratio is applied for load capacitance and supply voltage while clock

clock speed is scaled by 1/0.707. So the area of the chip is now 0.707^2A. If we calculate a new power per unit space, we have $0.707CX0.707V^2Xf/(0.707^3XA)$. Hence, the power per unit area becomes unchanged. But unfortunately, in 65nm technology and below, this law ceased because of exponential growth in leakage current and reduction in supply voltage decreases the speed of the processor. Nonetheless, the high performance demand is continued and this stimulates a shift from the single core to a multicore paradigm.

C. Need for Energy Saving Techniques

Today's innovations in semiconductor technology lead to not only an increase in the number of cores on a die, but also an increase in power density and concomitant heat dissipation. Increasing power dissipation leads many negative impacts on power delivery, performance/watt (PPW) ratio, packaging and cooling costs, reliability, availability and overall performance of the processors. So energy consumption issues occasionally more important than speed of the processor.

Energy efficiency is essential in mobile electronics where devices are battery powered. For last few decades, processor performance has been accelerating at a rate faster than the evolutions in battery technologies. This has led to a considerable drop of the battery life in mobile devices. At the same time, modern computational intensive applications demand very high performance. These two conflicting requirements, the need to conserve energy and the demand to deliver outstanding performance lead new approaches to resolve it. Existing researches to achieve optimal energy budget have two significant guidelines. First is in what way to increase the processor's efficiency within a specified energy limit. Second isin what way to decrease the energy consumption of computing devices without compromising processor performance.

III. OVERVIEW OF CLASSIC ENERGY SAVING TECHNIQUES

In this section, we delve into the up-to-date techniques in energy saving of MCAs. As of now, several hardware and software approaches have been adopted for alleviating power and energy costs. Through this investigation, we aim to demonstrate how the research community is trying to achieve an outstanding performance and energy efficiency. We can classify the power management techniques into three broad categories: Hardware techniques, Hardware-enabled middleware techniques and Software techniques.

A. Hardware techniques

Several energy saving techniques with dedicated controllers are embedded into the modern processor

architectures to provide energy efficiency. Applications running in a multicore domain need a carefully tailored computing architecture to meet their QoS within the power budget. The architectural innovations in designing of core, memory and interconnection networks improve the energy efficiency significantly.

1) Core Layout

Puttaswamy et al. propose a 3D microarchitecture with Thermal Herding techniques, which provides outstanding PPW ratio [18]. Compared to a conventional planar processor the proposed architecture can achieve 15% to 30% of active power reduction depending on the characteristics of the application. But 3D architecture incurs augmented power per unit area and associated temperature issues. This problem is resolved by Fazal Hameed et al. They present a thermal-aware 3D microarchitecture that effectively integrates the potential gains of dynamic architectural adaptation, fail-safe DVFS, and global migration [19]. Research shows that thermal-aware 3D architecture can achieve significant power reduction over 3D multicore processors [18] because it can reduce the active and the leakage powers simultaneously.

Kontorinis et al. introduce an adaptive processor with peak power guarantees which can reduce peak power by table-driven reconfiguration [20]. Most of the functional units (e.g. ALU, L1 cache, register files, load-store units and so forth) of the processor are dynamically organized for power conservation and maximum performance whereas peak power constraints are assured. Adaptive processor can reduce peak power about 25% with a smaller amount of performance cost.

Rodrigues et al. propose Dynamic Core Morphing (DCM) architecture for heterogeneous multicores [21]. The resources of the cores are morphed at runtime based on varying performance requirements. Depending on the computational need of the current workload, two cores may swap the execution units to maximize the PPW ratio.

The Scalable stochastic processor developed by Narayanan et al. has been demonstrated as an auspicious way to tackle power dissipation problem for error-tolerant applications such as audio or video. Improved scalability is realized by substituting or augmenting conventional computational units by gracefully degrading functional units [22]. The scalability leads power savings range between 20% and 60% in the well-known H.264 video encoder.

2) Memory Design

Many researches are carried out to bring innovations in memory organization in order to minimize the

power dissipation. Smart caching [23] emphases on power saving computing techniques and implements way-predicting caches with reduced leakage designing techniques. Flaunter et al. [24] explore the use of instruction pre-fetch algorithms combined with the drowsy caches, where cache lines are periodically put into a low power mode without considering their access histories. Implementation of a drowsy cache in a 0.07μm CMOS process can reduce 50% to 75% of the total energy consumption in the caches. Cai and Lu present a joint venture for saving energy in system memory and hard disk unanimously [25]. This technique periodically reconfigures the size of physical memory by adding or freeing up the allotted memory pages and uses a timeout policy for shutting down the hard disk. The suitable memory size and timeout are selected according to their proportionality with the average power consumption. This technique achieves energy savings higher than 50% over a fixed-timeout scheme.

3) Interconnection network

Several researches show that the design choices for interconnect fabric have significant impact on the power budget [20, 23]. The interconnect network itself is a power consuming resource. The power consumption of the interconnection network for a 16 core processor is more than the combined power consumption of two cores. Rakesh Kumar et al. [26]demonstrates the need for careful co-design of interconnect network and memory hierarchy. The power consumption of the core increases super linearly with the number of connected units and average length of wire. So a power-optimized architectural design needs compact length of interconnection wire segments and appropriate routing algorithms [23].

B. Hardware-Enabled Middleware techniques

The following techniques are employed as middleware and partially implemented in hardware. The hardware enables middleware to shut down or slow down the functional units according to the operating temperature. Hardware-enabled middleware techniques including Stop-and-Go [27], Dynamic Voltage and Frequency Scaling [29], Advanced Configuration and Power Interface [50] and different Gating Techniques [44,46] have attracted a great deal of attention.

1) Stop-and-Go

The stop-and-go is the simplest form of dynamic power management (DPM) technique [79]. The DPM techniques reduce the power consumption by shutting down or lowering the performance of idle cores. Stop-and-go can be realized on both global and local scale. In global approach, if one of the cores reaches its specified threshold temperature, this scheme shut down the whole chip until its non-critical

level has been recovered. If stop-and- go is realized locally, only the overheating core will be halted until it has cooled down. Global stop-and-go mechanism provides a smaller amount of control and less efficiency as a particular overheating core leads to unwanted delaying of all other non-critical cores.

Donald et al. [27] implement 12 combinations of local and global stop-and-go policies with other power management techniques (i.e. DVFS and Task migration) for managing the temperature of multicore processors. They investigate the pros and cons of each combination by comparing their performance. Whenever peak temperature of the processor reaches 84.2° C the stop-and-go controller shuts down the cores for 30msec to allow the cores to cool down. Their implementation results show that local stop-and-go can outperform global schemes. Chaparro et al. [28] propose a stop-and-go mechanism with clock gating mechanism. Whenever the core reaches its critical temperature, this combined technique halts the core, stores its current state information, and then shuts the core off completely. There is no dynamic as well as leakage power consumption in this technique. Soit allows the overheating core to cool down quicker. As the current state of the core is saved before shutting down, this method would be the ideal choice of the options.

2) Dynamic voltage and frequency scaling (DVFS) [29]

DVFS is the prevailing and powerful DPM technique, used to regulate the power consumption of the processor by dynamically scaling the level of supply voltage and clock frequency [29, 30, 31, 32]. The sub-threshold leakage current and gate-oxide tunneling leakage can be reduced by reducing supply voltage [33]. Reducing the clock frequency reduces the supply voltage linearly and decreases power consumption quadratically [34]. DVFS is widely used for memory bound workloads and can be employed in two ways:

1. The Local (per-core) DVFS allow us to scale the voltage of individual cores, so that the overheating core can cool down faster [28, 35, 36].
2. The Global DVFS allow us to adjust the voltages and frequencies of all cores uniformly and simultaneously. Similar to stop-and-go, a single hotspot on one of the cores could result to unnecessary performance penalty on all cores [28, 35, 37].

Using Local DVFS introduces more flexibility as each core can select its own voltage– frequency pair individually. However, that suffers from a large number of expensive inherent voltage regulators. The global DVFS can solve the thermal issues faster but the efficiency of DVFS is affected by limited flexibility to determining a single optimal voltage to all cores.

Weiser et al. present the first paper to suggest an interval based DVFS for reducing the power dissipation in computing devices. Their work focuses on three scheduling algorithms: Unbounded-delay perfect-future (OPT), Bounded-delay limited-future (FUTURE), and Bounded-delay limited-past (PAST) [38]. The deployment of each algorithm controls the clock frequency and makes the scheduling decisions simultaneously. The PAST scheduling algorithm with a 50msec adjustment interval can achieve power conservation of 50% to 70% based on circuit conditions.

Wonyoung et al. develop a fast, per-core DVFS mechanism with on-chip integrated voltage regulators [32]. This mechanism uses the potential benefit of both per-core voltage regulation and very fine-grained voltage switching. The in-built regulators can increase the energy efficiency opportunities of DVFS and result in 21% of energy savings over conventional global DVFS with off-chip regulators.

Many researchers have unified DVFS technique with thread migration polices to reduce energy consumption. Cai et al. develop a new thread shuffling algorithm, which integrates thread migration and DVFS techniques on a MCA supporting simultaneous multithreading (SMT) [39]. Thread shuffling dynamically migrates slower threads with same criticality degrees to a particular core and implements DVFS for other cores having fast threads. The proposed scheme realizes energy savings around 56% with no performance degradation. Quan Chen et al. propose an Energy-Efficient Workload Aware (EEWA) task scheduler, that consists of a work load aware frequency adjuster and a preference-based task-stealing scheduler [40]. With the help of DVFS, the workload-aware frequency adjuster can accurately configure the frequencies of the cores in an efficient fashion based on the profiled workload statistics. The preference-based task-stealing scheduler can successfully distribute the tasks across various cores at runtime according to the preference list. The EEWA can save energy about 28.6% with only 0.9% of performance loss.

All the previous works cited above, simply fail to consider the static power that has turn into a substantial portion of the total power consumption, unfortunately. LeSueur and Heiser [41+ assess the factors influencing the efficiency of DVFS on AMD Opteron processors, using an extremely memory-bound benchmark. They illustrate that the ability of DVFS is retreating in modern digital systems due to escalating leakage power. Furthermore, their investigation reveals that switching-off idle cores will facilitate greater energy savings. To reduce leakage power, Awan et al. [42] propose an enhanced race-to-halt (ERTH) approach. By integrating DVFS and slack management policies, ERTH can improve energy efficiency considerably.

When global DVFS is realized in MCA, determination of optimal voltage that satisfies all cores is a challenging endeavor; some applications will suffer from performance penalty or overheads. This issue exacerbates as the running applications and number of cores in next generation processors. From a hardware implementation point of view, local DVFS is more expensive than global DVFS, because of its costly inherent voltage regulators and phase- locked loops. However, the per-core DVFS provides better tradeoff between performance and power.

3) Gating Techniques

Clock Gating [44] and Power Gating [46] are very useful methods for decreasing dynamic and static power correspondingly [43]. Gating techniques are realized by insertion of anadditional logic between the clock source and clock input of the processor's circuitry. It diminishes power consumption by logically turning off (gating) the power to the portions of core that are not useful to the current workload.

a) Clock Gating (CG) Techniques

The clock gating techniques employed in the Hexagon™ Digital Signal Processors (DSP) are analyzed by Bassett et al. [44]. The proposed four levels of clock gating and spine-based clock distribution allow switching off the power to the different regions, from single logic cell to entire chip. Further power reduction is achieved through a structured clock tree by distributing the clock signal across the chip with low skew and delay. This technique provides reduction in power consumption by 8% for active mode and over 35% for sleep mode.

Hai et al. describe a deterministic clock gating (DCG) technique, which hinge on the advance knowledge about at what time the functional block will be idle in the upcoming cycles [45]. With this advance information DCG can switch-off the idle blocks that maximize the energy efficiency. By exploiting DCG to various functional units, the proposed technique achieves 19.9% of average diminution in power without any performance cost. However, for all these techniques, the effectiveness of gating is restricted by the granularity of components that can be gated, the failure to change the overall size and complexity of the processor. Also, these designs are still vulnerable to leakage inefficiencies.

b) Power Gating (PG) Techniques

Power Gating (PG) is a circuit-level technique to reduce leakage power consumption by effectively turning off the source voltage to the idle elements. PG can be applied either at the core-level [46] or at the unit-level of the processor such as cache banks, ALUs, pipeline branches etc. [47, 48]. Recently, Intel Core i7 processors use power gating transistors to turn off its idle cores [49].

Hu et al. develop a parameterized model based on analytical equations, which decides the breakeven point used for proper gating. They evaluate the dynamic power gating ability of the fpu (floating-point units) and fxu (fixed-point units) of POWER4 processor by three techniques namely ideal, time-based, and branch-misprediction-guided [47]. The implementation of these techniques in various execution units shows that a considerable decrease in static power consumption can be realized through power gating.Lungu et al. propose a Success Monitor Switch (SMS) and a Token Counting Guard Mechanisms (TCGM) for applying predictive power gating technique in POWER6 processor [48]. By employing SMS, the control logic enables or disables the PG depends on the success of policy. By implementing work for TCGM, this predictive power gating achieves a guarantee on the worst case execution of the policy. Leverich et al. [46] propose a Per-Core Power Gating (PCPG) technique with DVFS for datacenter workloads. This combined technique can save up to almost 60% of energy consumption.

4) Advanced Configuration and Power Interface (ACPI) [50]

ACPI is an industry standard for efficient handling of power management in computing devices. It is developed by the collaborative effort of Intel, Hewlett-Packard, Phoenix, Microsoft, and Toshiba [50]. ACPI provides platform-independent interfaces for power management and monitoring. These interfaces have the potential to work with existing DPM techniques [50]. ACPI is relay on operating system-directed configuration and Power Management (OSPM), which defines four switchable C-states (CPU idle states) C0, C1, C2, and C3 and n P-states (CPU-performance states) P0 to Pn for active power management. ACPI allows the processor to achieve fine tuning of the power consumption by moving idle devices into lower power states (sleeping state). Bircher and John [51] point out the implicit and explicit performance impacts of various CPU-idle states and Performance states of AMD quad-core processors. They verify their results for both compute-bound and memory-bound applications with fixed and OS scheduling. They develop an enhanced hardware and operating system configurations that decreases average active power by 30% with 3% of performance loss.

C. Software techniques

The performance per watt ratio of a MCA is depends on efficient built-in hardware and the ability of software to effectively control the hardware. Many up-to-date processors exploit software level power management techniques for energy efficiency. Recently, researchers have paid greater attention to the software power management policies because it can gain the power disparity statistics of processing threads on the fly with low cost. Software techniques can achieve predictable performance through transferring or scheduling tasks to minimize thermal gradients and hot spots. Software-based

approaches include data forwarding [52, 53] and task scheduling [54].

1) Data forwarding

Most modern processors use large size of on-chip L1 caches with multiple ports. Such a cache consumes a substantial part of the overall power owing to its larger size and high frequency access rate. Researches reveal that L1 data cache contributes 15% of the overall energy consumption of the processor [52]. Thus it is essential to develop tactics for precluding large power consumption in cache. Data forwarding is one of the appropriate solutions to reduce the energy consumption of L1 data cache.

Carazo et al. [53] propose a data cache filtering technique with forwarding predictor to reduce the power consumption of L1 data cache (DL1). This mechanism exploits an effective utilization of load-store queue (LSQ), which is responsible to provide the right data to load instructions by data forwarding method. Their experiments exhibit that the proposed cache filtering technique can achieve an average power savings up to 36% with a 0.1% of performance degradation. To reduce the access rate of DL1, Nicolaesu et al. propose a cached LSQ (CLSQ) to maintain load and store instructions after their execution [52]. Hitting in the CLSQ is faster and wastes not as much of energy as a DL1 access. Thus the significant savings in the frequency of accesses leads to 40% of energy reduction without any additional hardware complexity and performance penalty.

2) Task scheduling

Task scheduling is another breakthrough technique in power management arises from software approach. These algorithms are designed to solve temperature issues by distributing tasks among different cores. There have been wide ranges of literature put out on scheduling algorithms to achieve more processor utilization, better power conservation and more uniform power density without degrading the processor throughout. These algorithms schedule the tasks across cores based on predetermined temperature threshold. Work proposed by Hsin-Hao Chu and Yu-Chon Kao is a perfect example of how an adaptive thermal-aware multi-core task scheduling algorithm with multiple run-time controllers can mitigate the inter-core thermal costs and dynamic variations of task execution [54]. Implementations of run-time controllers increase the system complexity. To resolve this problem, the temperature-aware task scheduling algorithm, called Low Thermal Early Deadline First (LTEDF) is suggested by Wu et al. The LTEDF allocates tasks based on a novel History Coolest Neighborhood First allocation algorithm [55]. Simulation of the LTEDF algorithm demonstrates that it can satisfy the timing constraints for soft real time tasks and minimize the thermal consequences simultaneously.

Power aware task scheduling algorithms for MCAs can be classified into three types [56]: the global (dynamic binding) approaches [57], the partitioned (static binding) approaches [58] and the semi partitioned approaches [59, 60]. In the global approach, any core in the multicore system may execute any task. Global scheduling saves tasks in single priority- ordered queue, shared by all processors [57]. At every moment, the global scheduler chooses the highest-priority task for operation and the tasks are permitted to migrate between the processors. There are three types of priority assignment schemes for MCA: fixed-priority, [61, 58], dynamic priority [64] and Proportionate Fair (PFair) priority [65].

Fisher et al. develop a thermal-aware global scheduling algorithm for sporadic real-time tasks based on two priority assignment schemes, namely the global earliest-deadline-first (EDF) and the global deadline-monotonic (DM) [62]. The suggested schemes can substantially lessen the peak temperature around 30°C to 70°C as compared to load- balancing strategies.

Wang et al. develop a scheduling approach for hard real-time systems. They execute delay analysis for generic task arrivals using First-In-First-Out (FIFO) scheduling and Static-Priority (SP) scheduling with reactive speed control techniques [62]. But Andersson and Baruah prove that the fixed priority scheduling algorithms cannot achieve a utilization bound greater than 50% [58]. Some researchers handle this deficiency by PFair priority assignment. Baker addresses the aforementioned problem and demonstrates a schedulability test for preemptive deadline scheduling of periodic or sporadic real-time tasks [64]. Baruah and Shun-Shii Lin propose a new Pinfair algorithm that is very efficient in terms of runtime complexity and has a superior density threshold for a very large subclass of generalized pinwheel task systems [65]. Levin et al. propose a deadline partitioning algorithm, called DP- WRAP algorithm to handle sporadic task sets with arbitrary deadlines [66].

In the partitioned approaches, task set is partitioned and statically allocated to a designated processor. These task set are executed by existing scheduling algorithms and migration across core is not permitted. Fan et al. present a Partitioned Scheduling algorithm with Enhanced RBound (PSER) that exploits a flexible task set scaling technique and enhanced utilization bound for fixed-priority periodic real-time tasks [67]. This algorithm effectivelyimproves the schedulability of the system. They combine PSER with Harmonic Aware Partition Scheduling (HAPS) in [68], which converts the complete task set into harmonic set and takes the benefit of harmonic relationship between tasks to achieve increased utilization bound up to 100% [69].

Andersson demonstrates global PFair and partitioned static-priority scheduling on multiprocessors [70]. Guan et al. develop two separate fixed-priority scheduling algorithms for light tasks and heavy

tasks. The algorithm RM-TS/light (Rate monotonic-task set for light loads) can execute light task sets with sustainable parametric utilization bound and the RM- TS algorithm can perform any task set, whereas the utilization bound is lower than a specified limit [71].

Recently, a significant portion of semi-partitioned approaches [59, 60, 72, 73, 74, 75, 76] have been proposed to minimize energy expenditure in multicores. Semi- partitioned algorithms allocate most of the tasks to one particular processor. But, limited tasks (i.e. less than number of cores – 1) are partitioned into many subtasks and are allocated to various cores under some constraints.

Lakshmanan et al. introduce a Highest-priority task-splitting (HPTS) algorithm to enhance the utilization bound of partitioned deadline-monotonic scheduling algorithms (PDMS) from 50% to 60% on implicit deadline task sets [75]. They can obtain 88% of average utilization with very low migration overhead for randomly generated implicit-deadline task sets by extend this algorithm, which assigns the tasks in the decreasing order in terms of their size. Kato et al. [74] and Andersson et al [60] present real-time scheduling algorithms with high schedulability. Similar to partitioned scheduling, the proposed algorithms assign each task to a specific processor but can divide a task into two processors if there is not sufficient capacity remaining on a processor. The semi-partitioned approaches are more efficient as compared to the conventional global and partitioned approaches theoretically [75, 76, 77] and also suitable for practical implementations [73]. Zhang et al. show that the implementation complexity of semi-partitioned scheduling algorithm is relatively low. They investigate semi-partitioned approaches in the Linux OS and demonstrate their results on an Intel Core-i7 processor [78].

IV. CONCLUSIONS

Current innovations in semiconductor technology lead to not only an increase in the number of cores on a die, but also an increase in power density and concomitant heat dissipation. This adversely affects the system reliability and availability. Achieving high performance with low power consumption is imposing a new challenge on IC fabrication technology. This article presents various effective techniques for alleviating power dissipation of MCA and its classification based on their attributes. We accept as true that our review will help the researchers and architects to acquire ideas into the next-generation multicore processors and encourage them to endorse new energy efficient elucidations for fabricating competent architectures.

REFERENCES

1 Gordon E. Moore. Cramming more components onto integrated circuits. vol. 86(1); pp. 82 – 85, IEEE (1998)

2. Christian Martin.: Post-Dennard Scaling and the final years of Moore's Law consequences for the evolution of multicore-architectures (2014)

3. Intel Corp.: Intel core i7-940 processor. Intel Product Information, 2009 [Online]. Available: http://ark.intel.com/cpu.aspx?groupId=37148

4. Advanced Micro Devices Inc.: Key architectural features—AMD Phenom II processors. AMD Product Information, 2008 [Online]. Available: http://www.amd.com/usen/Processors/ProductInformation/ 0,30118153311591 7% 5E159 19, 00.html

5. Johnson, T., Nawathe, U.: An 8-core, 64-thread, 64-bit power efficient SPARC SoC (niagara2). In Proceedings of 2007 International Symposium on Physical Design (ISPD '07), pp. 2 – 2, ACM, (2007)

6. May, D.: XMOS XS1 architecture. Micro IEEE 2012; vol. 32(6); pp. 28 – 37.

7. ARM Ltd.: The ARM Cortex-A9 Processors. ARM Ltd. White Paper, Sept.2007 [Online] Available: http://www.arm.com/pdfs/ARM CortexA-9Processors .pdf

8. Advanced Micro Devices Inc.: ATI Radeon HD 4850 & ATI Radeon HD 4870—GPU specifications. AMD Product Information, 2008. [Online]Available: http://ati.amd.com/products/radeonhd4800/ specs3. html

9. NVIDIA Corp.: NVIDIA CUDA: Compute unified device architecture. NVidia CUDA Documentation, June 2008. [Online] Available: http://developer.download.nvidia.com/compute/cuda/2_0/docs/NVIDIA_CUDA_Pro *gramming_ Guide_2.0.pdf*

10. ARM Ltd.: ARM Development Tools, 2011. [Online] Available: http://www.arm.com/products/tools/development-boards/versatileexpr ess/index. php

11. Gschwind, M., Hofstee, H.P., Flachs, B., Hopkin, M., Watanabe, Y., Yamazaki, T.: Synergistic Processing in Cell's Multicore Architecture, vol. 26(2), pp.10 – 24, IEEE (2006)

12. Pericas, M., Cristal, A., Cazorla, F.J., Gonzalez, R., Jimenez, D.A., Valero, M.: A flexible heterogeneous multi-core architecture. In Proceedings of 16th International Conference on Parallel Architecture and Compilation Techniques, pp.13 – 24, IEEE (2007)

13. Zhuravlev, S., Saez, J.C., Blagodurov, S., Fedorova, A., Prieto, M.: Survey of Energy- Cognizant Scheduling Techniques, vol. 24(7), pp.1447 – 1464, IEEE (2013)

14. David Chinnery, Kurt Keutzer.: Overview of the Factors Affecting the Power Consumption. In proceeding of Tools and Techniques for Low Power Design, pp.11 – 53, Springer, (2007)

15. Dennard, R.H., Gaensslen, F. H., Yu, H., Rideout, V.L., Bassous , E., LeBlanc, A.R.: Design of ion-implanted MOSFET's with very small physical dimensions. In proceedings of IEEE Solid-State Circuits, vol. 12(1); pp. 38 – 50, IEEE (2007)

16. Hennessy, J.L., Pattersson, D.A.: Computer Architecture - A Quantitative Approach. Morgan Kaufmann, second edition (1996)

17. Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, Doug Burger.: Dark silicon and the end of multicore scaling. In proceeding of 38th International Symposium on Computer Architecture (ISCA), pp. 365 – 376, IEEE (2011)

18. Puttaswamy, K., Loh.: Thermal Herding: Microarchitecture Techniques for controlling hotspots in high-performance 3D integrated processors. In proceeding of 13thInternational Symposium on High Performance Computer Architecture, pp.193 – 204, IEEE (2007)

19. Fazal Hameed, Mohammad Abdullah Al Faruque, Jorg Henkel.: Dynamic thermal management in 3D multi-core architecture through run-time adaptation. In proceeding of Design, Automation & Test in Europe Conference & Exhibition, pp.1– 6, IEEE (2011)

20. Kontorinis, V., Shayan, A., Tullsen, D., Kumar, R.: Reducing Peak Power with a Table- Driven Adaptive Processor Core. In Proceedings of IEEE/ACM 42nd Annual International Symposium on Microarchitecture (MICRO-42), pp. 189 – 200, IEEE (2009)

21. Rodrigues, R., Annamalai, A., Koren, I., Kundu, S., Khan, O.: Performance per Watt Benefits of Dynamic Core Morphing in Asymmetric Multicores. In Proceedings of International Conference on Parallel Architectures and Compilation Techniques, pp. 121–13, ACM (2011)

22. Narayanan, S., Sartori, J., Kumar, R., Jones, D.: Scalable Stochastic Processors. In Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), pp.335 – 338, IEEE (2010)

23. Kornaros G.: Multi-core Embedded Systems. Taylor and Francis Group, CRC Press, (2010)

24. Flautner, K., Kim, N., Martin, S., Blaauw, D., Mudge, T.: Drowsy Caches: Simple Techniques for Reducing Leakage Power. In Proceedings of 29th Annual International Symposium on Computer Architecture, pp. 148 – 157, IEEE (2002)

25. Le Cai, Yung-Hsiang Lu.: Joint Power Management of Memory and Disk. In Proceedings of the conference on Design, Automation and Test in Europe, pp. 86 – 91, IEEE Computer Society (2005)

26. Kumar, R., Victor Zyuban, Dean M. Tullsen.: Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In Proceedings of 32nd International Symposium on Computer Architecture (ISCA), pp.408 – 419, IEEE (2005)
27. Donald, D., Martonosi, M.: Techniques for Multicore Thermal Management: Classification and New Exploration. In Proceedings of 33rd International symposium on Computer Architecture, pp.78 – 88, IEEE (2006)
28. Chaparro, P., Gonzalez, J., Magklis, G., Cai, Q. Gonzalez, A.: Understanding the Thermal Implications of Multicore Archtectures. IEEE Transactions on Parallel and Distributed Systems, vol.18 (8), pp. 1055 – 1065, IEEE (2007)
29. Isci, C., Buyuktosunoglu, A., C.-Y. Chen, Bose, P., Martonosi, M.: An Analysis of Efficient Multi-Core Global Power Management Policies: Maximizing Performance for a Given Power Budget. In Proceedings of 39th Annual IEEE/ACM International Symposium on MICRO-39, pp. 347 – 358, IEEE (2006)
30. Hanumaiah, V., Vrudhula, S.: Energy-efficient Operation of Multicore Processors by DVFS, Task Migration and Active Cooling. Computers, vol. 63, no. 2, pp. 349 – 360, IEEE (2012)
31. B. de Abreu Silva, Bonato V.: Power/performance optimization in FPGA-based asymmetric multi-core systems. In Proceedings of 22nd International Conference on Field Programmable Logic and Applications (FPL), pp. 473 – 474, IEEE (2012)
32. Wonyoung Kim, Gupta M S, Gu-Yeon Wei, Brooks D.: System level analysis of fast, per-core DVFS using on-chip switching regulators. In Proceedings of 14th International Symposium on High Performance Computer Architecture, pp.123 – 134, IEEE (2008)
33. Dongwoo Lee, Wesley Kwong, David Blaauw, Dennis Sylvester.: Simultaneous Subthreshold and Gate-Oxide Tunneling Leakage Current Analysis in Nanometer CMOS Design. In Proceedings of 4th International Symposium on Quality Electronic Design, pp. 287 – 292, IEEE (2003)
34. Burd, T., Pering, T., Stratakos, A., Brodersen, R.: A dynamic voltage scaled microprocessor system. In Proceedings of International Solid-State Circuits Conference, pp. 294 – 295, IEEE (2000)
35. Jayaseelan, R., Mitra, T.: A Hybrid Local-global Approach for Multi-core Thermal Management. In Proceedings of 2009 IEEE/ACM International Conference on Computer-Aided Design, pp. 314 – 320, IEEE (2009)
36. Pruhs, K., Van Stee, R., Uthaisombut, P.: Speed scaling of tasks with precedence constraints in approximation and online algorithms. In Proceedings of 3rd International conference on approximation and online algorithms, pp.307– 319, Springer (2006)
37. March, J.L., Sahuquillo, J., Hassan, H., Petit, S., Duato, J.: A new energy-aware dynamic task set partitioning algorithm for soft and hard embedded real-time systems. vol. 54, no. 8, pp. 1282 – 1294, The Computer Journal (2011)
38. Weiser, M., Welch, B., Demers, A., Shenker, S.: Scheduling for reduced CPU energy. In Proceedings of 1st USENIX conference on OSDI, pp. 13 – 23, ACM (1994)
39. Cai Qiong, Gonzalez, J., Magklis, G., Chaparro, P., Gonzalez, A.Q.:Thread shuffling: Combining DVFS and thread migration to reduce energy consumptions for multi-core systems. In Proceedings of 2011 International Symposium on Low Power Electronics and Design, pp. 379 – 384, IEEE (2011)
40. Quan Chen, Long Zheng, Minyi Guo, Zhiyi Huang.: EEWA: Energy-Efficient Workload- Aware Task Scheduling in Multi-core Architectures. In Proceedings of Parallel & Distributed Processing Symposium Workshops (IPDPSW), pp.642 – 651, IEEE (2014)
41. Le Sueur E., Heiser G.: Dynamic voltage and frequency scaling: The laws of diminishing return. In Proceedings of Hot Power: Workshop on Power aware computing and systems, pp. 1– 8 (2010)
42. Awan, M.A, Petters, S.M.: Enhanced Race-To-Halt: A Leakage-Aware Energy Management Approach for Dynamic Priority Systems. In Proceedings of 23rd EUROMICRO Conference on Real-Time Systems (ECRTS), pp.92 – 101, IEEE (2011)
43. Li Li, Ken Choi, Haiqing Nan.: Activity-driven fine-grained clock gating and run time power gating integration, vol. 21(8); pp. 1540 – 1544 IEEE (2013)
44. Bassett, P., Saint-Laurent M.: Energy efficient design techniques for a digital signal processor. In Proceedings of International Conference on IC Design & Technology, pp.1–4, IEEE (2012)
45. Hai Li, Swarup Bhunia, Yiran Chen, Vijaykumar T N, Roy K.: Deterministic Clock Gating for Microprocessor Power Reduction. In Proceedings of 9th International Symposium on High-Performance Computer Architecture, pp.113 – 122, IEEE (2003)
46. Leverich, J., Monchiero, M., Talwar, V., Ranganathan, P.: Power management of data center workloads using per-core power gating. vol. 8(2); pp. 48 – 51, IEEE (2009)
47. Hu, Z., Buyuktosunoglu, A., Srinivasan, V., Zyuban, V., Jacobson Bose, P.: Micro architectural Techniques for Power Gating of Execution Units. In Proceedings of 2004 International Symposium on Low Power Electronics and Design, pp.32 – 37, IEEE (2004)

48. Lungu, A., Bose, P., Buyuktosunoglu, A., Sorin, D.: Dynamic Power Gating with Quality Guarantees. In Proceedings of International Symposium on Low Power Electronics and Design (ISLPED), pp. 377– 382, IEEE (2009)
49. Rajesh Kumar, Glenn Hinton.: A Family of 45nm IA Processors. In Proceedings of IEEE International Solid-State Circuits Conference, pp. 58 – 59, IEEE (2009)
50. Hewlett-Packard, Intel, Microsoft, Phoenix Technologies, Toshiba: Advanced Configuration and Power Interface Specification, Revision 4.0a. April 2010, [Online]. Available: http://www.acpi.info/spec.html.
51. Lloyd Bircher, Lizy, W., John, K.: Analysis of Dynamic Power Management on Multi- Core Processors. In Proceedings of 22nd Annual International Conference on Supercomputing, pp. 327–338, ACM (2008)
52. Dan Nicolaescu, Alex Veidenbaum, Alex Nicolau: Reducing Data Cache Energy Consumption via Cached Load/Store Queue. In Proceedings of 2003 International Symposium on Low Power Electronics and Design, pp. 252 – 257, IEEE (2003)
53. Carazo, P., Apolloni, R., Castro, F., Chaver, D., Pinuel, L., Tirado F.: L1 Data cache power reduction using a forwarding predictor, vol. 6448; pp.116–125, Springer (2011)
54. Hsin-Hao Chu, Yu-Chon Kao, Ya-Shu Chen.: Adaptive thermal-aware task scheduling for multi-core systems. vol. 99; pp.155–174, ELSEVIER (2015)
55. Wu, G., Xu, Z., Xia, Q., Ren, J., Xia, F.: Task allocation and migration algorithm for temperature-constrained real-time multi-core systems. In Proceedings of 2010 IEEE/ACM International Conference on Green Computing and Communications, pp.189–196, IEEE (2010)
56. Carpenter, J., Funk, S., Holman, P., Srinivasan, A., Anderson, J., Baruah, S.: A Categorization of Real-time Multiprocessor Scheduling Problems and Algorithms. In Handbook on Scheduling Algorithms, Methods, and Models, pp. 30.1– 30.19 (2006)
57. Andersson, B.: Global Static-Priority Preemptive Multiprocessor Scheduling with Utilization Bound 38%. In Proceedings of ACM International Conference on Principles of Distributed Systems (OPODIS), vol. 5401; pp.73–88, ACM (2008)
58. Andersson, B., Baruah, S., Jonsson, J.: Static-Priority Scheduling on Multiprocessors. In Proceedings of 2nd Real-Time Systems Symposium, pp.193 – 202, IEEE (2001)
59. Kato, S., Yamasaki, N.: Semi-partitioned fixed-priority scheduling on multiprocessors. In Proceedings of 15th Real-Time and Embedded Technology and Applications Symposium, pp.23 – 32, IEEE (2009)
60. Andersson, B., Bletsas, K, Baruah, S.: Scheduling Arbitrary Deadline Sporadic Task Systems on Multiprocessors. In Proceedings of Real-Time Systems Symposium, pp. 385 – 394, IEEE (2008)
61. Lundberg, L.: Analyzing fixed-priority global multiprocessor scheduling. In Proceedings of 8th Real-Time and Embedded Technology Symposium, pp.145–153, IEEE (2002)
62. Fisher, N., J.-J. Chen, Wang, S. Thiele, L.: Thermal aware global real-time scheduling on multicore systems. In Proceedings of Real-Time and Embedded Technology and Applications Symposium, pp. 131–140, IEEE (2009)
63. Wang, S., Bettati, R.: Delay analysis in temperature constrained hard real-time systems with general task arrivals. In Proceedings of 27th IEEE International Real- Time Systems Symposium, pp. 323–334, IEEE (2006)
64. Baker, T.P.: An analysis of EDF schedulability on a multiprocessor. vol. 18(8); pp. 760
– 768, IEEE (2005)
65. Baruah, S.K, Shun-Shii Lin: Pfair scheduling of generalized pinwheel task systems. Transactions on Computers, vol.47 (7), pp. 812– 816, IEEE (1998)
66. Levin, G., Funk, S., Sadowski, C., Pye, I., Brandt, S.: DP-fair: A simple model for understanding optimal multiprocessor scheduling. In Proceedings of 22nd EUROMICRO Conference, pp. 313, IEEE (2010)
67. Ming Fan, Qiushi Han, Gang Quan, Shangping Ren: Multi-core partitioned scheduling for fixed-priority periodic real-time tasks with enhanced RBound. In Proceedings of 15th International Symposium on Quality Electronic Design, pp.284 – 291, IEEE (2014)
68. Ming Fan, Qiushi Han, Shuo Liu, Shaolei Ren, Gang Quan, Shangping Ren: Enhanced fixed-priority real-time scheduling on multi-core platforms by exploiting task period relationship. pp.85– 96, Elsevier (2014)
69. Liu, J.W.S.: Real-time systems. Prentice Hall (2000)
70. Andersson, B., Jonsson, J.: The utilization bounds of partitioned and pfair static priority scheduling on multiprocessors are 50%. In Proceedings of 15th EUROMICRO Conference on Real-time Systems, pp.33 – 40, IEEE (2003)
71. Guan, N., Martin Stigge, Wang Yi, Ge Yu: Parametric Utilization Bounds for Fixed- Priority Multiprocessor Scheduling. In Proceedings of 26th International Parallel and Distributed Processing Symposium, pp.261-272, IEEE (2012)
72. Anderson, J.H., Bud, V., Devi, U.C.: An EDF-Based Scheduling Algorithm for Multiprocessor Soft Real-Time Systems. In Proceedings of EUROMICRO Conference on Real-Time Systems (ECRTS), pp.199– 208, IEEE (2005)
73. Bastoni, A., Brandenburg, B.B, Anderson, J.H.: Is Semi-partitioned scheduling practical? In Proceedings of 23rd Conference on Real-Time Systems, pp. 125 – 135, IEEE, (2011)
74. Kato, S., Yamasaki, N.: Semi-partitioned fixed-priority scheduling on multiprocessors. In Proceedings of 15th Real-Time and Embedded Technology and Applications Symposium, pp. 23 – 32, IEEE (2009)

75. Lakshmanan, K., Rajkumar, R., Lehoczky, J.P.: Partitioned fixed priority preemptive scheduling for multi-core processors. In Proceedings of 21st EUROMICRO Conference on Real-Time Systems, pp. 239 –248, IEEE (2009)
76. Guan, N., Stigge, M., Yi, W., Yu G.: Fixed-Priority Multiprocessor Scheduling with Liu and Layland's Utilization Bound. In Proceedings of IEEE Real Time and Embedded Technology and Applications Symposium (RTAS), pp. 165 – 174, IEEE (2010)
77. Guan, N., Martin Stigge, Wang Yi, Ge Yu: Fixed-Priority Multiprocessor Scheduling: Beyond Liu and Layland's Utilization Bound. In Proceedings of WiP Real-Time Systems Symposium (RTSS), pp. 1594– 1601, IEEE, (2010)
78. Zhang, Y., Guan, N., Yi. W.: Towards the Implementation and Evaluation of Semi- Partitioned Multi-Core Scheduling. In Bringing Theory to Practice: Predictability and Performance in Embedded Systems. vol. 18; pp. 42– 46, In Open Access Series in Informatics (2011)
79. Young-Si Hwang, Ki-Seok Chung: Dynamic Power Management Technique for Multicore Based Embedded Mobile Devices. IEEE Transactions on Industrial Informatics, vol. 9(3), pp. 1601 – 1612 (2013)

Model Reference Adaptive Control (Mrac) Scheme For Eliminating Over Shoot In Dc Servomotor

[1]Okafor Patrick U, [2]Eneh Princewill Chigozie, [3]Arinze S. N.
[1,2,3] Department of Electrical and Electronic Engineering Department,
Faculty of Engineering,
Enugu State University of Science and Technology (ESUT), Enugu State, Nigeria

A B S T R A C T

This paper presents a way of eliminating overshoot in a DC servomotor using a model reference adaptive control system, which is a controller in the Neural Network (NN) toolbox. A Speed and Position based MRAC system and a Proportional Integral Derivative (PID) controller were used to control a physical DC servomotor with known parameters. This was achieved through physical measurements by configuration of the servo driver, the computer system and the DC motor using, Simulink toolbox and Microsoft Windows Software Development kits (SDKs) 7. The reference input to the system was a step function input. The purpose is to determine the time-response of the developed system also, the stability of the system and its ability to reach one stationary state when starting from another. Results from physical measurements show that MRAC system was able to accommodate nonlinearities associated with DC motor and yet, maintain good control of the motor without voltage overshoot as against the PID controller.

Keywords—Artificial Neural Network (ANN), ANN Inverse Model (AIM), DC Motor , Model Reference Adaptive Control (MRAC), Overshoot, Proportional Integral Derivative (PID)

1.0 INTRODUCTION

The most common actuator in control systems is a Direct current (DC) motor, apparently the choice for implementation of advanced control algorithms in most electric drives. Lately, developments in magnetic materials, microprocessors, semiconductor technology and mechatronics etc. had provided a wide range of applications for high performance electric motors in various industrial and automated processes. For high performance driveapplications such as robotics, machine tools, conveyors, rolling mills, etc., high precision in speed and position control is paramount. DC servomotors are mostly the choice; they are widely deployed in these applications due to their reliability and ease of control because of the decoupled nature of the field and the armature magneto motive force (Sheel, Chandkishor, and Gupta, 2010).

DC Servomotor systems have two outputs that can be controlled, angular speed and angular position. Improving the system's efficiency comes from the proper control of both outputs together, or by

controlling one of them at a time and, hence, two identification plants can be derived for each one. For some applications, such as disk drives and robotics, position control is more important than speed control (Makableh, 2011). One of the parameters that negatively affect the efficient control of DC servomotor is overshoot. In the context of control theory, overshoot can be regarded as an output exceeding its final steady state value. Overshoot could be seen as a form of distortion that affects the rise time, settling time etc. of an operating plant under step input response. Reviews show that conventional controllers such as Proportional Integral Derivative (PID) are not that capable of handling nonlinearities associated with DC motors and at the same time, mitigate the effects of overshoot.

Thus, one of the drawbacks of conventional tracking controllers for electric drives is that they are unable to capture the unknown load characteristics over a widely ranging operating point. Apparently, this makes tuning of controller parameters very difficult. There are many ways to overcome these difficulties but, generally there are four basic way that are common to adaptive controller; (1) Model reference adaptive control (MRAC), (2) Self tuning, (3) Dual control and (4) Gain scheduling. Usually load torque is a nonlinear function of a combination of variables such as speed and position of the rotor. Therefore, identifying the overall nonlinear system through a linearized model around a widely varying or changing operating point, under fast switching frequencies, can introduce errors which can lead to unstable or inaccurate performance of the system (Astron and Wittenmark, 1989).

THEORY

Basic Operation of DC Motors

The dc motor is a torque transducer that converts electric energy into mechanical energy. It has the electrical and the mechanical representation. The torque developed on the motorshaft is directly proportional to the field flux and the armature current. To substantiate that, assume a current-carrying conductor is established in a magnetic field with flux Φ, and the conductor is located at a distance r from the center of rotation. The relationship among the developed torque, flux Φ, and current i_a is

$$T_m = k_m \phi i_a. \qquad (1a)$$

Where;

T_m= the motor torque (in N-M), Φ, = the magnetic flux in (in webers), i_a = the armature current (in amperes), k_m= a proportional constant. In addition to the torque developed, when the conductor moves in the magnetic field, a voltage is generated across its terminals. This voltage is known as the *back emf*, which is proportional to the shaft velocity, and tends to oppose the current flow. The relationship between the back emf and the shaft velocity is:

$$T_m = k_m \phi i_a. \qquad (1a)$$

Where;

e_b= the back emf (in volts), w_m= the shaft velocity of the motor (in rad/sec). Equations 1a and 1b form the fundamentals of the dc-motor operation.

Model Reference Adaptive Control (MRAC)

MRAC is an adaptive servo system in which the desired performance is expressed in terms of a reference model, which gives desired response to the reference signal. Generally speaking, MRAC is composed of four parts namely; the plant containing unknown parameters, a reference model for compactly specifying the desired output of the control system, a feedback control law containing adjustable parameters. The adaptation law of MRAC systems extracts parameter information from the tracking errors. The online computation of this controller, like NARMA-L2, is minimal. However, unlike NARMA-L2, the model reference architecture requires that a separate neural network controller be trained offline, in addition to the neural network plant model. The controller training is computationally expensive, because it requires the use of dynamic backpropagation (Beale, Hagan, and Demuth, 2015). On the positive side, model reference control applies to a larger class of plant than does NARMA-L2 control. Fig. 1 shows an illustration of a neural network (NN) MRAC system.

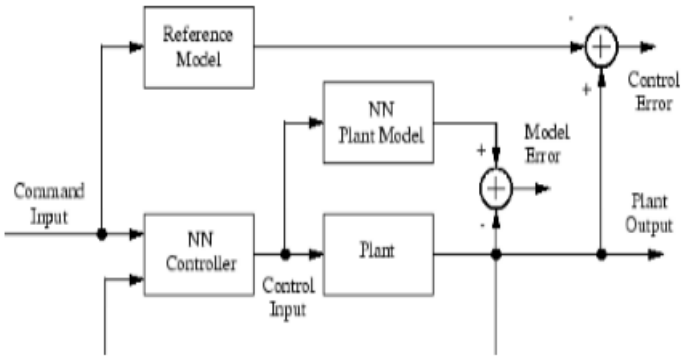


Fig 1 Block diagram of Model Reference Adaptive Control

Recently, artificial neural networks represent an effective alternative for industrial process modeling and control mainly because they can learn nonlinear input/output mapping from process data (Narendra and Parthasarathy, 1990). Perhaps, it has been emphasized that complete controllability and observability of the process must be assumed for successful neural network modeling and control (Saerens and Soquet,1991). More so Narendra and Parthasarathy (1990) indicated that considerable

progress in nonlinear control theory is still needed to obtain rigorous solutions to identification and control problems using neural networks.

PID Control Systems

The PID controller is a form of close loop system i.e. with a feedback mechanism. The PID control system is achieved by tuning and calculating the error signal between an output measured value and a reference value (input), the controller works to minimize the error signal or the difference between the output signal and the reference signal to a minimum value; such that the output measured value will be as close as possible to the input reference signal.

The mathematical model of the PID controller has been proposed by many authors and is represented by:

$$U(t) = Kp.e(t) + Ki\int_0^t e(\tau)d\tau + Kd\frac{d}{dt}e(t) \quad (2)$$

Where

$U(t)$ is the controller output signal, $e(t)$ i is the error signal, Kp is the proportional gain, Ki is the integral gain and Kd is the derivative gain.

Modeling the Permanent DC Motor

DC motors are used extensively in control systems especially in industrial actuators so, it is paramount to establish mathematical model for analytical purposes for efficient control application of dc motors. The DC motor takes in single input in the form of an input voltage and generates a single output parameter in the form of output speed. It is a single-input, single-output system (SISO). Figure 2 is the electromechanical representation of a DC motor, the diagram is used to develop the system level transfer function that characterize the operation or behavior of a DC motor.

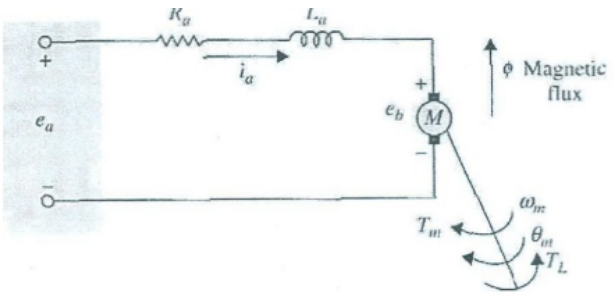


Fig 2 Electrical Model of DC Motor

The armature is modeled as a circuit with resistance R_a connected in series with an inductance L_a and a voltage source e_a, and e_b representing the back electromotive force (emf) in the armature when the rotor rotates. Looking at the diagram of fig 1, it can be seen that the control of the dc motor is applied at the armature terminals in the form of applied voltage e_a (t). It can be deduced that the torque developed in the motor is proportional to the air-gap flux and the armature current. The equations that describes the DC servomotor behavior as stated by (Kuo and Golnaraghi (2002) thus,

$T_m(t) = K_m(t)\,\phi\, i_a(t)$ (3)

Since ϕ is constant, equation 3 is in form

$T_m(t) = K_i\, i_a(t)$

$K_i\, i_a(t) = i_m\omega_m + b\omega_m + T_L$ (4)

Where:

$T_m(t)$=motor torque. $i_a(t)$ = armature current. $T_L(t)$ = load torque. Φ= magnetic flux in the air gap. k_m = proportionality constant. K_i = torque constant in N-m/A. $\omega_m(t)$=rotor angular velocity i_m = equivalent moment of inertia reflected at the motor shaft . Putting the control input voltage e_a (t) into consideration, the cause and effect equations for the motor circuit in same fig 4 are:

$$\frac{di_a(t)}{dt} = \frac{1}{L_a}e_a(t) - \frac{R_a}{L_a}i_a(t) - \frac{1}{L_a}e_b(t). \quad (5)$$

$$e_b(t) = k_b\frac{d\theta_m(t)}{dt} = K_b\omega_m(t). \quad (6)$$

$$\frac{d^2\theta_m(t)}{dt^2} = \frac{1}{J_m}T_m(t) - \frac{1}{J_m}T_L(t) - \frac{B_m}{J_m}\frac{d\theta_m(t)}{dt} \quad (7)$$

Where:

$l_a(t)$ = armature inductance

R_a = armature resistance. $e_a(t)$ =applied voltage

$e_b(t)$ = back emf. K_b = back emf constant

$\omega_m(t)$= rotor angular velocity. B_m = viscous-friction coefficient. θ (t)= rotor displacement

J_m=rotor inertia. From equations 3 through 6, the applied voltage $e_a(t)$ is considered as the cause and Equation 5 considers that $\frac{di_a(t)}{dt}$ the immediate effect due to the applied

voltage. From Equation 3, armature current $i_a(t)$ causes the motor torque $T_m(t)$, while in Equation 6 the back emf $e_b(t)$ was defined. It can be seen also from Equation 7 that the motor torque produced causes the angular velocity $\omega_m(t)$ and displacement θ_m (t) of the rotor respectively.

The state variables are of the system can be define as

- Armature current = $i_a(t)$
- Rotor angular velocity = $\omega_m(t)$
- Rotor angular displacement = θ_m (t)

It is possible to eliminate all the non-state variables from Equation 3 through 7 by direct substitution then present the dc state equation in vector-matrix form as follows:

$$\begin{bmatrix} \frac{di_a}{dt}(t) \\ \frac{d\omega_m(t)}{dt} \\ \frac{d\theta_m(t)}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} & -\frac{K_b}{L_a} & 0 \\ \frac{K_i}{J_m} & -\frac{B_m}{J_m} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} i_a(t) \\ \omega_m(t) \\ \theta_m(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_a} \\ 0 \\ 0 \end{bmatrix} e_a(t) + \begin{bmatrix} 0 \\ -\frac{1}{J_m} \\ 0 \end{bmatrix} T_L(t). \quad (8)$$

Note that in the case of Equation 8 above, that $T_L(t)$ is handled as a second input to the state equations. The transfer function between the motor displacement and the input voltage is obtained as thus;

$$\frac{\Theta_m(s)}{E_a(s)} = \frac{K_i}{L_a J_m s^3 + (R_a J_m + B_m L_a) s^2 + (K_b K_i + R_a B_m) s} \quad (9)$$

Note that T_L has been set to zero in Equation 9. Fig 2 shows a block diagram of the DC motor system for speed control. From the diagram, one can see clearly how the transfer function is related to each block. It can be seen from Equation 9 that s can be factored out of the denominator and the significance of the transfer function $\frac{\Theta_m(s)}{E_a(s)}$ is that the dc motor is an integrating device between these two variables. $\Theta_m(s)$ is the rotor angular displacement Laplace transfer function, $E_a(s)$ is the input voltage Laplace transfer function and Ωm(s) is
the transform of angular velocity respectively. From fig 3 also, it can be seen that the motor has a built-in feedback loop caused by the back emf Eb.

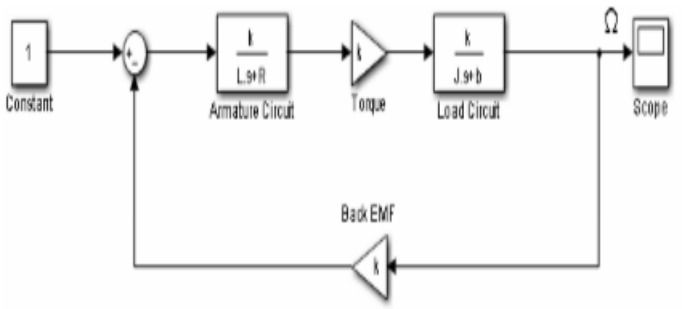


Fig 3 Simulink Model of a DC Servomotor in terms of speed

The back-emf physical represents the feedback of a signal that is proportional to the negative of the speed of the motor. From equation 9, it can be noted that back emf constant K_b represents an added term to the resistance R_a and the viscous-friction coefficient B_m . Effectively, the back-emf effect is equivalent to an electric friction which tends to improve the stability of the motor and apparently the stability of the system.

3.2. DC Motor Equivalent Circuit In Discrete Form

Recall that ANN is the modeling tool. So simulation can be performed on the control of DC motor using ANN model, there is need to construct an equivalent DC motor to a discrete time model. Effectively, the load torque is assumed as:

$$T_L = \mu\omega^2\, m(t)[sgn(\omega_m\,(t))] \qquad (10)$$

Where μ = a constant

It is obvious that from equation (10) that load torque is always opposes the direction of motion. Note that the choice of load torque here is arbitral because considering load torque as one of the functions of a DC motor; it is a common characteristic for most propeller driven loads.

Alternatively,

Direct substitution and substitute for position in equations (4), (5) and (10) i.e. rewriting the angular velocity $\omega_m(k)$, which is the speed in terms of angular displacement θ_m which is the position as shown in figure 4.

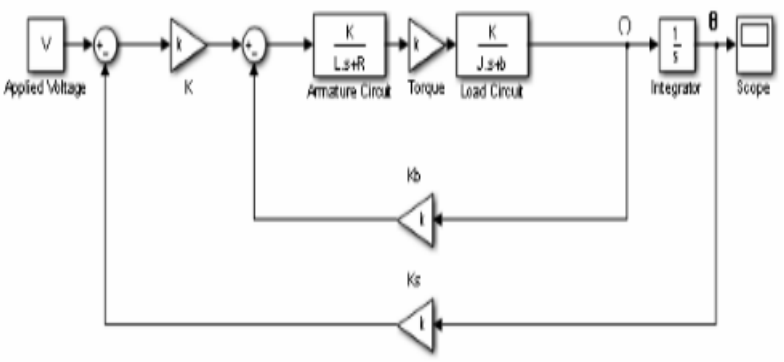


Fig 4 Simulink Model of a DC Servomotor in terms of Speed and Position

Deploying direct substitution and substitute for position in equations (4), (5) and (10) i.e. rewriting the angular velocity $\omega_m(k)$, which is the speed in terms of angular displacement θ_m which is the position as shown in figure 3. Then the equations yields;

$$k_e\theta_m = -R_a i_a - L_a \frac{di_a}{dt} + v_a. \qquad (11)$$

$$k_e i_a = I_m\theta_m + b\theta_m + T_L \qquad (12)$$

$$T_L = \mu \left(\frac{d\theta_m}{dt}\right)^2 [sgn(\theta_m)]. \qquad (13)$$

Next is to estimate the derivatives of position and current in discrete form using a sampling interval of ΔT and forward difference.

$$\frac{d\theta_m(k)}{dt} = \frac{\theta_m(k+1) - \theta_m(k)}{\Delta T}. \qquad (14)$$

$$\frac{d^2\theta_m(k)}{dt^2} = \frac{\frac{d\theta_m(k+1)}{dt} - \frac{d\theta_m(k)}{dt}}{\Delta T}. \qquad (15)$$

$$\frac{d^3\theta_m(k)}{dt^3} = \frac{\frac{d^2\theta_m(k+1)}{dt^2} - \frac{d^2\theta_m(k)}{dt^2}}{\Delta T} \qquad (16)$$

$$\frac{di_a}{dt} = \frac{i_a(k+1) - i_a(k)}{\Delta T}. \qquad (17)$$

$$T_L(k) = \mu \left(\frac{d\theta_m(k)}{dt}\right)^2 [sgn(\theta_m(k)]. \qquad (18)$$

$$\frac{dT_L(k)}{dt} = \frac{T_L(k+1) - T_L(k)}{\Delta T}. \quad (19)$$

Next is to evaluate the armature current i_a in terms angular displacement θ_m which is the position here. Then substituting i_a in terms of θ_m using equations (11) into the following equation $\frac{K_e{}^2(\Delta T)^2 + R_a(b\Delta)^2}{(L_a I_m + R_a I_m \Delta T)}$ from the work of Weerasooriya and El-Sharkawi (1991), to determine the function governing the speed control of a DC motor, gives

$$k_e\dot{\theta}_m = -\frac{R_a}{k_e}\left[I_m\ddot{\theta}_m + b\dot{\theta}_m + T_L\right] - \frac{L_a}{k_e}\left[I_m\dddot{\theta}_m + b\ddot{\theta}_m + \dot{T}_L\right] + v_a. \quad (20)$$

Or

$$v_a = \frac{L_a I_m}{k_e}\dddot{\theta}_m + \left[\frac{R_a I_m}{k_e} + \frac{L_a b}{k_e}\right]\ddot{\theta}_m + \left[\frac{R_a b}{k_e} + k_e\right]\dot{\theta}_m + \left[\frac{R_a}{k_e}T_L + \frac{L_a}{k_e}\dot{T}_L\right] \quad (21)$$

Integrating equations (12), (13), (14), (15), (16) and (17) into equation (19) then the input voltage in equation (19) can be written as a function of:

$\theta_m(k+1)$.

$\theta_m(k)$.

$\theta_m(k-1)$.

$\theta_m(k-2)$.

$$v_a(k) = g[\theta_m(k+1), \theta_m(k), \theta_m(k-1), \theta_m(k-2)] \quad \textbf{(22)}$$

Effectively, equation (22) forms the relationship between the input voltage v_a and the motor position θ_m at four successive sampling instances. Assume that the term $\theta_m(k+1)$ is replaced in equation (22) with desired reference motor position at next instance as $\theta_d(k+1)$, and compute the control voltage (the input voltage) $v_a(k)$ with the following equation, then

$$v_a(k) = g[\theta_d(k+1), \theta_m(k), \theta_m(k-1), \theta_m(k-2)]. \quad \textbf{(23)}$$

So, if the computed voltage $v_a(k)$ at sampling instance *k* is applied, then the resulting motor position at instant $(k+1)$ will be equal to:

$\theta_m(k+1) = \theta_d(k+1)$, i.e. the desired motor position, it effectively takes the following forms of input to the ANN

$\theta_d(k+1)$. = the reference position

$\theta_m(k)$. = Position at first instance

$\theta_m(k-1)$. = Position at second instance

$\theta_m(k-2)$. Position at third instance

Structure of the ANN controller

The non-linear controller for this work is the Artificial Neural Network (ANN) Controller. Here, the design of ANN incorporated a Feed forward Neural Network (FFNN), which is made up of one input layer and one hidden layers with an output layer. The layers respectively consist of number of neurons. Each neuron has two functions as:

- Summing up all the outputs from the previous layers multiply by the corresponding weights
- Performing the nonlinear sigmoidal or linear function on the sum

During training, errors are back propagated and also minimized using least mean square algorithm. The basis for weights connection between the input and hidden layers are based on the fact that errors in the output determine the measures of the hidden layer output errors. This adjustment of weights between the layers and recalculating the output in an iterative process is continued till the error falls below a tolerable level.

Training the ANN Controller

The training data was chosen taking into consideration the mechanical and hardware limitations of the motor. And that lead to the following constrained operating range as defined below (Weerasooriya and El-Sharkawi,1991):

$\omega(k)-\omega(k-1)<0.1$

$\omega(k)<30.0$ rad/s

$v(k)<100.0$v

The corresponding target voltage was then generated by using these speed values. The offline training of the controller is performed with this training data using the back- propagation algorithm. The back-propagation training algorithm is based on the principle of minimization of a cost function of the outputs and the target of the FFNN.

The input and output of this network is guided by some basic equations, the net input of the j^{th} neuron of the hidden layer at the time instant n is given as follows (Haykins, 1999):

$$S_i^h(n) = \sum_{I=1}^{N} W_{ii}^h(n) I_i(n). \qquad (24)$$

Where W_{ij}^h is the connecting weight between the i^{th} neuron at the input layer and the j^{th} neuron at the hidden layer. The I_i is the i^{th} input, and N is the number of inputs. Then, the output from the j^{th} neuron from the hidden layer at n^{th} instant is given by:

$$O_j^h(n) = f^h[S_j^h(n) + B_j^h(n)] \qquad (25)$$

From equation 25, B_j^h is the bias of the j^{th} neuron and f^h is the activation function acting on each neuron at the hidden layer. The activation function can be tan sigmoidal, log sigmoidal or linear. The functions are described as follows (Beale, Hagan and Demuth, 2015):

$$tansig(x) = \frac{1-e^{-2x}}{1+e^{-x}} \qquad (26)$$

$$logsig(x) = \frac{1}{1+e^{-x}} \qquad (27)$$

$$linear(x) = x \qquad (28)$$

In the above equations x represents the input to the activation function. It follows that the net input of the k^{th} neuron of the output layer at time instant n is given by:

$$S_k^o(n) = \sum_{i=1}^{M} w_{ik}^o(n)\, O_i^h(n) \qquad (29)$$

Where M is the number of neurons in the hidden layer and $w_{jk}^o(n)$ is the weight between the j^{th} neuron at the hidden layer and k^{th} neuron at the output layer respectively. It

therefore also followed that output from the k^{th} neuron at the output layer at time instant n can be presented in the form:

$$O_k^0(n) = f^0[S_k^0(n) + B_k^o(n)] \qquad (30)$$

Where f^0 = the activation function of the output layer and

$B_k^o(n)$ = the bias of the k^{th} neuron at the output layer.

It is also important to put into consideration how the weight is updated at various levels during the network training. To do that, there is a basic equation that describes the updating of the weight through the error signal at the output of the neuron *k* at the iteration and it is given as follow:

$$e_k(n) = d_k(n) - O_k^o(n) \qquad (31)$$

Where $d_k(n)$ represents the desired output for neuron *k*.

ANN MODEL OF DC MOTOR

Before ANN can be used to control the operation of the DC motor, the DC motor being the plant must also be modeled using ANN tool. From equation 23 where $v_a(k)$ is a function of speed at successive time intervals k+1, k and k-1 for any required trajectory, what happens is that the ANN Inverse Model (AIM) generates an output that is proportional to the voltage required at the input of the DC motor to produce these speed at the time intervals. Here, the output-input mapping is many to one perhaps, disturbances and other uncertainties may lead to the input-output mapping to become one-to-many leading to degradation in the control performance. Though, the AIM relies on the accuracy of the model used for the controller design so, the research will not worry about the degradation. The block diagram of the speed based AIM is shown in fig 5.

Fig 5 Block Diagram of The AIM

Structure of The AIM

The AIM for this research work is made up of three inputs and a single output structure for the three successive speed instances. Based on equation 23, the three inputs are $\omega_m(k+1)$; Speed at first instance $\omega_m(k)$; speed at second instance, $\omega_m(k-1)$; speed at third instance and the output is the $V_a(k)$ which is the motor terminal output voltage $V_t(k)$ from fig 6.

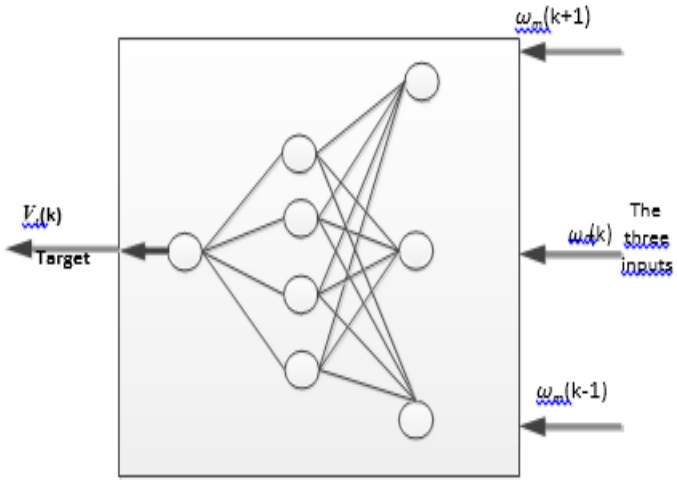


Fig 6 the structure of AIM

So based on the same equation 23, the nonlinear function (f.) can be presented in the following form:

$$f(\omega_m(k+1),\omega_m(k),\omega_m(k-1)) = \frac{(\omega_m(k+1)-\alpha\omega_m(k)-\beta\omega_m(k-1)-\gamma\, sgn(\omega_m(k))\omega_m{}^2(k)-\delta sgn(\omega_m(k-1))\omega_m{}^2(k-1)}{\zeta} \quad (32)$$

The values of $(\omega_m(k+1), \omega_m(k), and\ \omega_m(k-1)$ apparently form the independent inputs of the ANN and the corresponding output as well is generated from equation 28.

Evaluating The Performance of The AIM

The generated $(\omega_m(k+1), \omega_m(k), \omega_m(k-1)$ inputs and the corresponding targets $V_a(k)$ are used for offline training of the AIM to represent any DC servomotor with unknown parameters. From figure 6, it could be seen that the performance error is represented by $e_i(k)$. In evaluating the AIM performance, the value of $[\ e_i(k)]^2$ for all kT that are elements of time from 0 to t_f is minimized, that is;

$$[e_i(k)]^2 \forall kT \in [0, t_f] \qquad (33)$$

Where T is the sampling period and t_f is the time for which simulation is performed. The terminal voltage (estimated) is given by:

$$\hat{V}_t = N[\omega_m(k), \omega_m(k-1), \omega_m(k-2)] \qquad (34)$$

Once the DC motor is excited by an input signal, the output from the DC motor which is speed in this context is fed into the AIM as an input. The terminal voltage i.e $\hat{V}_t(k-1)$ is compared with the actual motor output ei(k) for a common excitation signal. Then the mean square value of the error ei(k) between the actual motor input and the estimated output voltage yields the performance error of the AIM.

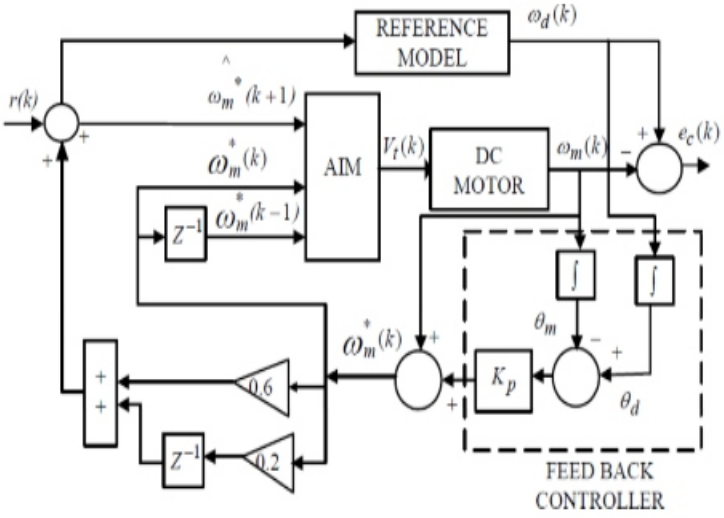


Fig 7 Structure of Speed and Position based MRAC Control System

6.0 THE PID CONTROLLER MODEL

The PID controller was achieved using equation 2 to get the best possible gains. Table 1 shows the DC motor parameters.

Table 1: DC Motor Parameter Values (Weerasooriya and El-Sharkawi, 1991)

Parameters	Values
Moment of Inertia J	0.064kg-m^2
Damping Coefficient b	0.03475Nm/s
Torque Constant Kt	3.475Nm A^{-1}
Electromotive force Constant Ke	3.40NmA^{-1}
Electrical Resistance R_a	7.56Ω
Electrical Inductance L_a	0.055H

The neural network controller is a speed and position based MRAC system as shown in fig 7. Figure 8 shows the construction of the PID controller in Simulink.

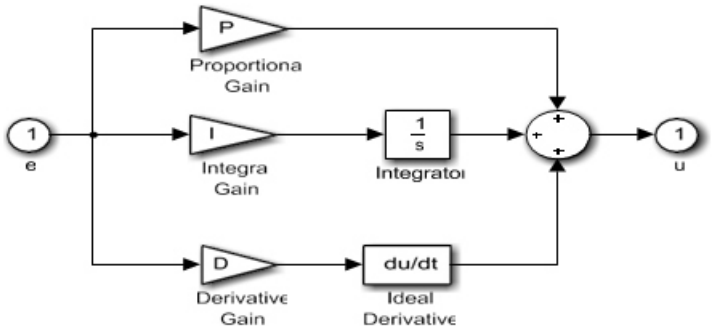


Fig 8 The PID Simulink Model

The PID controller is incorporated in the DC motor Simulink model to determine the effect of the PID controller on the system as shown in figure 9.

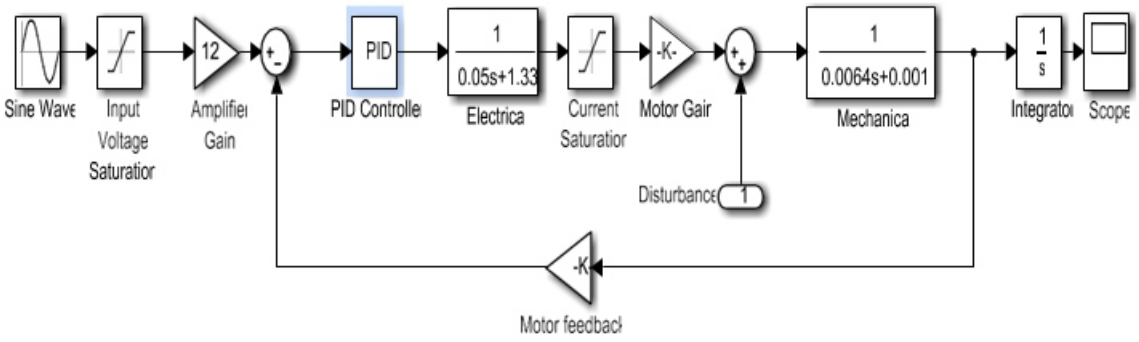


Fig 9: DC Motor Simulink Model with PID Controller

Measurement and Results

The Speed and Position based MRAC system and the PID controller system were used to drive a physical DC motor. The reference input to the system is a step function input. The essence of this is to determine the stability of the systems and their ability to reach one stationary state when starting from another. Step input value of 180 (degrees) were chosen arbitrary and used as input to both MRAC control system and the PID controller. The output time responses were graphed for both controllers to determine the existence of overshoot in voltage as shown in figures 9 and 10.

It could be observed from figure 9, that the Speed and Position based MRAC system was able to achieved steady state at about 0.56 sec without overshoot while the PID controller from figure 10, achieved steady state at about 2.41sec with voltage overshoot.

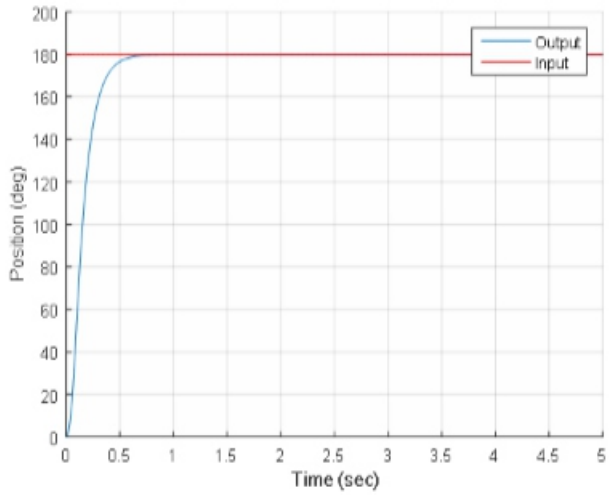


Fig 10 Step Input Response of Speed And Position Based MRAC System (180 Degrees)

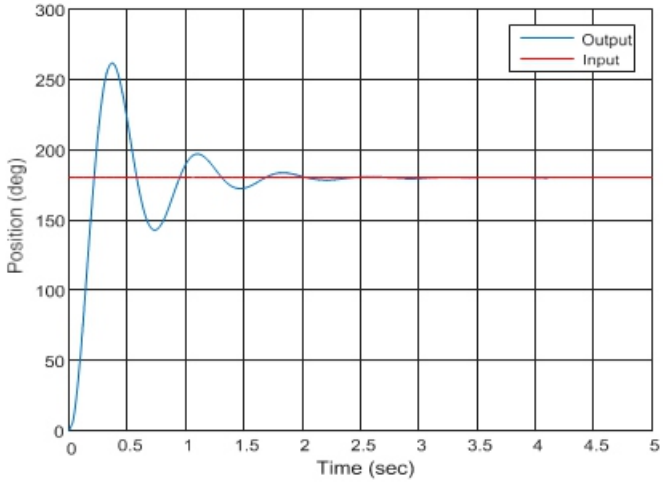


Fig 11 Step Input Response of PID Controller (180 Degrees)

CONCLUSION

Results from measurements show that the conventional PID controller was not able to accommodate the nonlinearities associated with the DC servomotor which lead to the distortion in the input voltage to cause overshoot between rise time and settling time. Perhaps, the speed and position based controller which is adaptive in nature, was able to accommodate those nonlinearities and yet maintain good control of the DC motor without voltage overshoot.

REFERENCES

1. Astrom K.J. and Wittenmark B., (1998). Computer Controlled Systems, Prentice-Hall.

2. Beale M.H, Hagan M.T. and Demuth H.B. (2015) Neural Network Toolbox User'

3. Haykins S. (1999) "Neural Networks– A comprehensive foundation", Second Edition, Prentice Hall, 1999

4. Kuo B. C and Golnaraghi F. (2002). Automatic Control Systems, Eight Edition. Pulisher: John Wiley & Sons, Inc. ISBN-13 978-0470-04896-2,

5. Makableh Y. (2011). Efficient Control of DC Servomotor Systems Using Backpropagation Neural Networks.Electronic Theses & Dissertations. 771. http://digitalcommons.georgiasouthern.edu/etd/771

6. Narendra K. S. and Parthasarathy K. (1990). "Identification and control of dynamical systems using neural networks", IEEE Transactions on Neural Networks, 1, (1), pp. 4- 27.

7. Saerens M. and Soquet A. (1991). "Neural Controller Based on Back Propagation Algorithm", IEEE Proceedings on Radar and Signal Processing, 138, (1), pp. 55 – 62.

8. Sheel S., Chandkishor R., and Gupta O. (2010). Speed Control of DC Drives Using MRAC Technique.International Conference on Mechanical and Electrical Technology:EEE.

9. Weerasooriya S. and El-Sharkawi M.A. (1991) "Identification and control of a DC motor using Back-propagation Neural Networks", IEEE Transactions on Energy Conversion, Vol.6, No.4, pp. 663-669.

Efficacy Of Artificial Neural Network For Financial Literacy Prediction

[1]Meenakshi Sood, [2]Puneet Bhushan

[1]Assistant Professor, Department of Electronics and Communication Engineering, Jaypee University of Information Technology Waknaghat, Solan, Himachal Pradeshl

[2]Assistant Professor, University Business School Himachal Pradesh University, H.P.

A B S T R A C T

Financial forecasting is becoming more and more dependent on advanced computer techniques due to high non-linearity and high volatility nature of finance domains. An Artificial Neural Network (ANN) is the current technique being used that can model flexible linear or non-linear relationship among variables and predict financial data more accurately. Financial Literacy (FL) is a combination of attitude, knowledge, skill and behavior to achieve desired financial goals. FL shows non linear dependencies on various socio demographic attributes which are in turn responsible for FL level of an individual. This paper brings neural networks applications in the financial domain. An attempt has been made in this research work to analyze the usefulness of artificial neural network for predicting FL level of individuals. The classification accuracy of 75% has been achieved through MLPNN that can help in knowing the efficacy of demographic determinants. The studies done in this work show that neural network have great promise for financial applications.

Keywords—Financial Literacy; Artificial Neural network; Sensitivity; ROC.

I. INTRODUCTION

Financial literacy is the knowledge regarding personal financial matters. Financial literacy enables individuals to handle their personal finances effectively and efficiently [1]. Thus financially literate individuals feel more satisfied and confident in decisions relating to money matters. The ability to handle personal finances is must as it helps in achieving individuals financial well being. Financial literacy has gained much importance in the recent past on account of changing financial landscape. These days many new innovative financial products are available in the market as investment options. Most individuals are not willing to invest in such products due to lack of knowledge regarding such products. More chances of getting misled on financial matters also increase if an individual is not financially literate. Thus minimum acceptable level of FL is necessary for individuals. Past research has pointed out that level of FL is low across the world. Thus the need of the hour is to improve the level of financial literacy across the world. The level of FL can be increased by way of imparting financial education to the population. Financial education programs are effective only if it becomes clear which section of population are the most financially illiterate [2]. The prediction regarding the level of FL

make from this model can help policy makers, academicians and researchers to identify the target population with low levels of FL. Accordingly, financial education programs can be targeted to those sections of population which are identified as being financially illiterate.

Authors in [3] have measured the financial literacy level of salaried individuals in India. They studied the determinants of FL. Their results showed that level of FL gets affected by gender, education, income, nature of employment and place of work whereas it does not get affected by age and geographic region. The model used in this paper also considers the socio- demographic variables which are used by authors in [3].

Recently, computer technologies have gain popularity and prominence as a potential tool in solving problems related to finance. One of the contemporary methods is Artificial Neural Networks (ANNs) that are good at classification, forecasting and recognition which make them good candidates for financial forecasting and for decision making process [4]. An ANN has to be trained and then tested such that the application of a set of inputs produces the desired set of outputs. Training involves feeding of teaching patterns to the network and defining a learning rule with a set of desired responses. A forward pass is done, and the errors or discrepancies between the desired and actual response for each node in the output layer are found [5]. These networks have self-learning capability and are fault-tolerant as well as noise immune.

In literature, several studies have been conducted who emphasize the Nonlinear Model proving to be more effective for finance forecasting. For this reason, ANN applications have been widely used in a variety of areas in financial markets. Authors in [6] have examined the current trends of applications of neural networks in finance. They have exhaustively compared the common characteristics of these applications with applications based on statistical and econometrics models.

This paper presents the use of machine learning in predicting the level off FL possessed by an individual based on demographic and socio-economic factors. To predict the level of FL, an Artificial Intelligence Model was implemented using ANN.

The rest of the paper is organized as follows. Section 2 describes the methodology adopted and includes a description of the procedure used to frame the demographic attributes random samples. Section 3 presents empirical results, the performance achieved and discussion of these results followed by conclusion in Section 4.

2. METHODOLOGY

The development of Financial Literacy Prediction (FLP) model is a two stage prediction system. The first stage is feature extraction and selection and second stage comprises of prediction technique, employing Artificial Neural network model. The flow diagram of the proposed methodology is depicted in Figure 1.

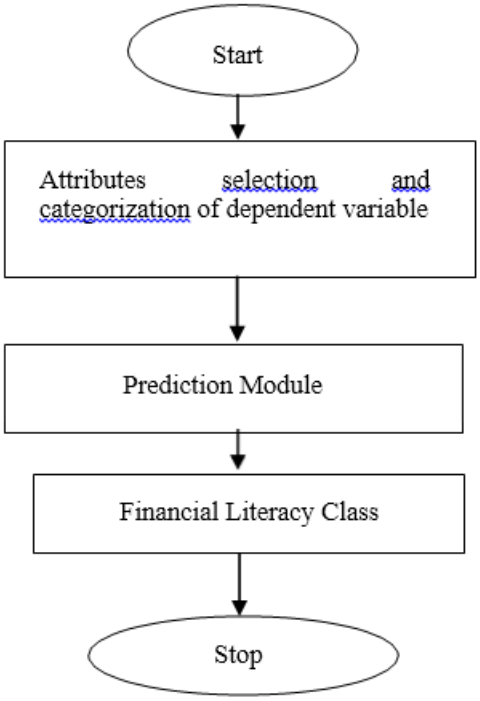


Fig 1. The structural flow diagram of the proposed FLP model Dataset

For carrying out this study, primary data was collected with the help of a questionnaire. The questionnaire was divided into two parts. First part collected personal information of the respondents relating to various socio-demographic factors. The second part consists of multiple choice questions in order to check the FL level of respondents. The study was confined to the state of Himachal Pradesh. Due to large geographic area, multistage samplinghas been used. The population of the study was salaried individuals working in the state of Himachal Pradesh. There are total of twelve districts in Himachal Pradesh out of which three districts were selected randomly. Further two sub-divisions from each selected districts were selected randomly. Finally respondents from these sub-divisions have been selected conveniently so that sample is representative of the population. The sample size of 516 is used for this study.

Respondents were asked thirteen multiple choice questions in various areas of risk, return, savings and investments, borrowings and financial planning. Based on the correct answers, FL score was calculated

for each respondent. We have created a dichotomous dependent variable and named it as Financial Literacy Level. The respondents whose scores were either equal to or below the median score, were put in one category labeled as low financial literacy. The respondents whose scores were more than the median value were put in another category labeled as high financial literacy. The methodology adopted for creating the database for the research work is depicted in Fig. 2.

3. RESULTS AND DISCUSSION

Model Selection

Structure of the neural network and network configuration depends on the number of hidden layers, number of neurons in each hidden layer, number of input neurons and the selection of activation function. The architecture of ANN is mostly problem dependent.

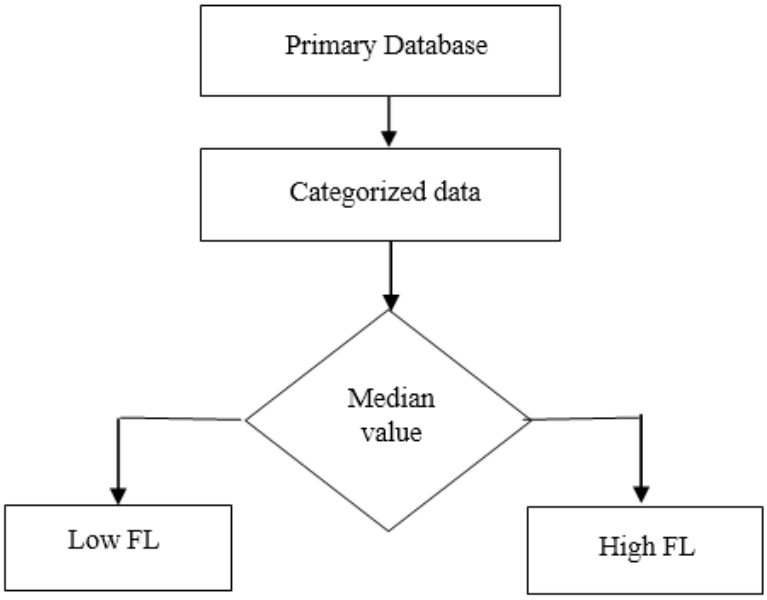


Fig 2. The flow diagram for categorizing FL score

Neural networks are capable of solving complex and non linear problems by learning and by being trained. Networks learn by getting exposed significant features of difficult problem. The best model was constructed after trying different topologies and different parameter settings, as there are no hard and fast rules in determining the best ANN architecture [7].

The FLPNN model consist of an ANN with set of attributes selected in the datset as the input, one hidden layer with eight hidden neurons, and two output neuron. The activation function chosen is hyperbolic tangent function in the hidden layer and softmax in the output layer. The error function employed at the output layer is cross-entropy. In total 516 samples of population were taken, out of which 71 % of the

samples (= 367 people) were used for training the Neural network and remaining 29% samples (= 149 persons) data were used for testing the network using bootstrapping method with 1000 seed points. No samples were excluded and none of them were missing. The learning rate of 0.07 is chosen after testing the net with values ranging from 0 to 1, as small values of learning rates are always preferable. The graphical representation of the FLP neural network is depicted in Fig. 3 on the last page because of space constraint.

PERFORMANCE METRICS

A set of performance metrics is used to provide a more detail analysis of the prediction model. Sensitivity, Selectivity and Accuracy, are the three key parameters used for the performance evaluation and validity of the proposed method [8]. These values are tabulated in the form of confusion matrix. The values obtained for this model are summarized in Table

1.

$$Specificity = \frac{TrueNegative}{TrueNegative + FalsePositive}$$

$$Sensitivity = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$Accuracy = \frac{TrueNegative + TruePositive}{All}$$

Confusion matrix is tabular description of classification of cases in the test dataset. In confusion matrix, the columns denote the actual cases and the rows denote the predicted class. Both the sensitivity and specificity measure classification accuracy against the ground truth diagnosis and are defined by Sensitivity is an assessment of the two upper quadrants while specificity is represented in the lower two quadrants. They are both bounced between 0 and 100%, and higher values indicated better accuracy. FLPNN succeeded in predicting Financial Literacy of individuals based on various socio demographic attributes. with the total accuracy of 75%. The empirical results show that proposed FLP model exhibits good sensitivity and specificity.

ROC curve

ROC curve is graphical representation of classification of test cases. It represents all the cases, actual and predicted on the plots with sensitivity (true positive rate) represented on the Y-axis and (1-specificity) (false positive rate) on X-axis. Area under the ROC curve is preferred to evaluate the performance of a classifier and its value lies between 0 and 1. The classifier has a perfect discriminating

ability if area of the ROC curve is 1, and classifier is unable to discriminate between two classes if area under the curve is 0.5[9]. The ROC curve obtained for FLP model using ANN is plotted shown in Fig 4. Area unnder ROC curve is 80% which is an indicator of good discrimination capability of FLPNN model.

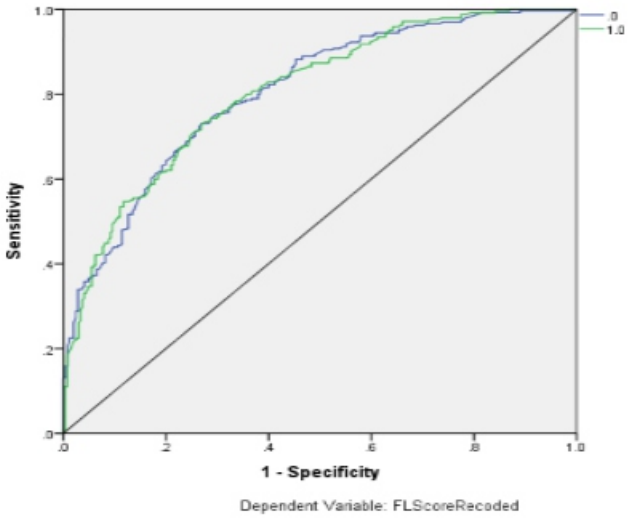


Fig 4. ROC curve of the FLP neural network model

Table 1. Confusion Matrix for classification

Sample	Observed FL	Predicted FL		
		0	1	Percent Correct
Training	0	134	65	67.30%
	1	41	127	75.60%
	Overall Percent	47.70%	52.30%	71.10%
Testing	0	56	16	77.80%
	1	21	56	72.70%
	Overall Percent	51.70%	48.30%	75.20%

CONCLUSION

It is almost impossible to do an exhaustive search for the best ANN model to be conclusive. The objective of this work is to design and implement an automatic prediction system of financial literacy level of individuals based on various demographic and socioeconomic factors. The authors have laid emphasis on selection of relevant attributes those are less related, balanced and converge to best solution. Authors have analyzed the applications of neural networks in finance and have concluded that neural networks have the capability of forecasting financial performance at the least cost. The uniqueness of the proposed methodology is that given the characteristics of various socio-demographic variables we are able to predict the level of financial literacy of individuals. The results obtained are

clear indicators of the great potential of neural networks in the areas of finance.

REFERENCES

[1] *Noctor, M., Stoney, S., & Stradling, R. Financial literacy: a discussion of concepts and competences of financial literacy and opportunities for its introduction into young people's learning, National Foundation for Education Research, 1992.*

[2] *Beal, D. J., & Delpachitra, S. B. "Financial literacy among Australian university students", Economic Papers,2003, 22 ,65-78.*

[3] *P Bhushan, Y Medury, "Financial literacy and its determinants", International Journal of Engineering, Business and Enterprise Applications, vol 4, no.2, 2013, pp. 155-160*

[4] *Meenakshi Sood and Sunil V. Bhooshan, "Parameter-selective based CAD system for epileptic seizure classification", International Journal of Applied Engineering Research, July 2015, vol 10, no.10, pp. 25389 - 25408.*

[5] *Anupam Tarsauliya et al. , " Analysis of Artificial Neural Network for Financial Time Series Forecasting "International Journal of Computer Applications (0975 – 8887), vol. 9, No5, November 2010 pp 16-22.*

[6] *Marzi, H.; Turnbull, M.; Marzi, E.; , "Use of neural networks in forecasting financial market," Soft Computing in Industrial Applications, 2008. SMCia '08. IEEE Conference on , 25-27 June 2008 pp.240-245,*

[7] *Fishman, M., Barr, D. S. and Loick, W. J., Using Neural Nets in Market Analysis,Technical Analysis of Stocks & Commodities, pp. 18–20, April 1991*

[8] *K. Kumar and S. Bhattacharya, "Artificial Neural Network vs Linear Discriminant Analysis in Credit Ratings Forecast: A Comparative Study of Prediction Performances," Review of Accounting and Finance, vol.5, no. 3, pp.217-227, 2006.*

[9] *Meenakshi Sood and Sunil V. Bhooshan 'A novel module based approach for classifying epileptic seizures using EEG signals", 2016 International Conference on Industrial Informatics and Computer Systems (CIICS), American University of Sharjah, Sharjah, March 13-15, 2016 pp.1-5.*

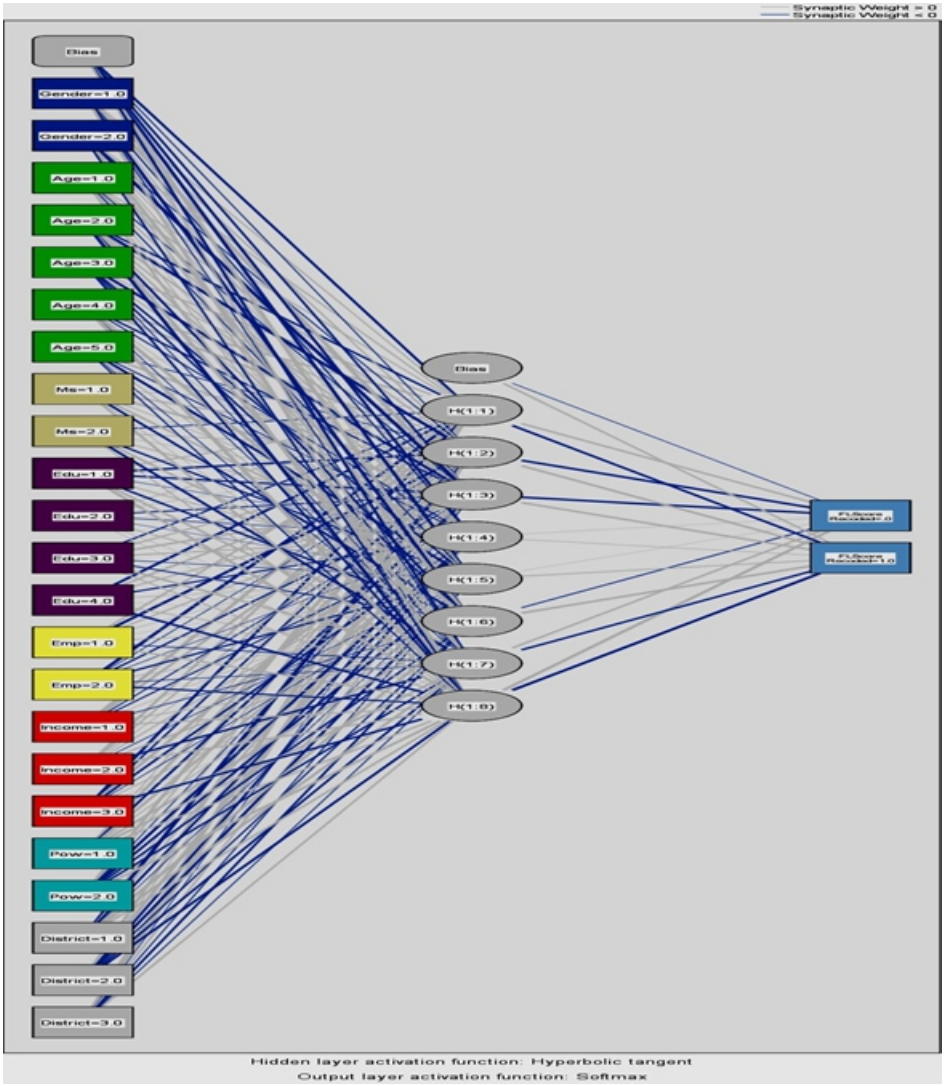


Fig. 3: FLP neural network architecture

Instructions for Authors

Essentials for Publishing in this Journal

1 Submitted articles should not have been previously published or be currently under consideration for publication elsewhere.

2 Conference papers may only be submitted if the paper has been completely re-written (taken to mean more than 50%) and the author has cleared any necessary permission with the copyright owner if it has been previously copyrighted.

3 All our articles are refereed through a double-blind process.

4 All authors must declare they have read and agreed to the content of the submitted article and must sign a declaration correspond to the originality of the article.

Submission Process

All articles for this journal must be submitted using our online submissions system. http://enrichedpub.com/ . Please use the Submit Your Article link in the Author Service area.

-

Manuscript Guidelines

The instructions to authors about the article preparation for publication in the Manuscripts are submitted online, through the e-Ur (Electronic editing) system, developed by **Enriched Publications Pvt. Ltd**. The article should contain the abstract with keywords, introduction, body, conclusion, references and the summary in English language (without heading and subheading enumeration). The article length should not exceed 16 pages of A4 paper format.

Title

The title should be informative. It is in both Journal's and author's best interest to use terms suitable. For indexing and word search. If there are no such terms in the title, the author is strongly advised to add a subtitle. The title should be given in English as well. The titles precede the abstract and the summary in an appropriate language.

Letterhead Title

The letterhead title is given at a top of each page for easier identification of article copies in an Electronic form in particular. It contains the author's surname and first name initial .article title, journal title and collation (year, volume, and issue, first and last page). The journal and article titles can be given in a shortened form.

Author's Name

Full name(s) of author(s) should be used. It is advisable to give the middle initial. Names are given in their original form.

Contact Details

The postal address or the e-mail address of the author (usually of the first one if there are more Authors) is given in the footnote at the bottom of the first page.

Type of Articles

Classification of articles is a duty of the editorial staff and is of special importance. Referees and the members of the editorial staff, or section editors, can propose a category, but the editor-in-chief has the sole responsibility for their classification. Journal articles are classified as follows:

Scientific articles:

1. Original scientific paper (giving the previously unpublished results of the author's own research based on management methods).

2. Survey paper (giving an original, detailed and critical view of a research problem or an area to which the author has made a contribution visible through his self-citation);

3. Short or preliminary communication (original management paper of full format but of a smaller extent or of a preliminary character);

4. Scientific critique or forum (discussion on a particular scientific topic, based exclusively on management argumentation) and commentaries. Exceptionally, in particular areas, a scientific paper in the Journal can be in a form of a monograph or a critical edition of scientific data (historical, archival, lexicographic, bibliographic, data survey, etc.) which were unknown or hardly accessible for scientific research.

Professional articles:

1. Professional paper (contribution offering experience useful for improvement of professional practice but not necessarily based on scientific methods);
2. Informative contribution (editorial, commentary, etc.);
3. Review (of a book, software, case study, scientific event, etc.)

Language

The article should be in English. The grammar and style of the article should be of good quality. The systematized text should be without abbreviations (except standard ones). All measurements must be in SI units. The sequence of formulae is denoted in Arabic numerals in parentheses on the right-hand side.

Abstract and Summary

An abstract is a concise informative presentation of the article content for fast and accurate Evaluation of its relevance. It is both in the Editorial Office's and the author's best interest for an abstract to contain terms often used for indexing and article search. The abstract describes the purpose of the study and the methods, outlines the findings and state the conclusions. A 100- to 250-Word abstract should be placed between the title and the keywords with the body text to follow. Besides an abstract are advised to have a summary in English, at the end of the article, after the Reference list. The summary should be structured and long up to 1/10 of the article length (it is more extensive than the abstract).

Keywords

Keywords are terms or phrases showing adequately the article content for indexing and search purposes. They should be allocated heaving in mind widely accepted international sources (index, dictionary or thesaurus), such as the Web of Science keyword list for science in general. The higher their usage frequency is the better. Up to 10 keywords immediately follow the abstract and the summary, in respective languages.

Acknowledgements

The name and the number of the project or programmed within which the article was realized is given in a separate note at the bottom of the first page together with the name of the institution which financially supported the project or programmed.

Tables and Illustrations

All the captions should be in the original language as well as in English, together with the texts in illustrations if possible. Tables are typed in the same style as the text and are denoted by numerals at the top. Photographs and drawings, placed appropriately in the text, should be clear, precise and suitable for reproduction. Drawings should be created in Word or Corel.

Citation in the Text

Citation in the text must be uniform. When citing references in the text, use the reference number set in square brackets from the Reference list at the end of the article.

Footnotes

Footnotes are given at the bottom of the page with the text they refer to. They can contain less relevant details, additional explanations or used sources (e.g. scientific material, manuals). They cannot replace the cited literature.
The article should be accompanied with a cover letter with the information about the author(s): surname, middle initial, first name, and citizen personal number, rank, title, e-mail address, and affiliation address, home address including municipality, phone number in the office and at home (or a mobile phone number). The cover letter should state the type of the article and tell which illustrations are original and which are not.

Address of the Editorial Office:

Enriched Publications Pvt. Ltd.
S-9,IInd FLOOR, MLU POCKET,
MANISH ABHINAV PLAZA-II, ABOVE FEDERAL BANK,
PLOT NO-5, SECTOR -5, DWARKA, NEW DELHI, INDIA-110075,
PHONE: - + (91)-(11)-45525005