ISSN:0975-9646

Implementing Synchronous Counter using Data Mining Techniques

Sangeetha S

Assistant Professor,Department of Computer Science and Engineering,
B.N.M Institute of Technology,
Bangalore, Karnataka, India

***Abstract*— Large amount of data are now available in business, industry, science and many other areas, due to rapid advances in computerization and digitization such data may provide a rich resource in knowledge discovery and decision support. In order to understand, analyse and eventually make use of huge amount of data, a multidisciplinary approach, a data mining, the process of identifying interesting patterns from large databases is proposed to meet the challenge. This paper consists of two proposed schemes which describes about how to derive synchronous counter without using any hardware components. That is, the paper tells about the conversion of hardware to software implementation using data mining techniques. One proposed method is CBDMRT (counter based data mining rule Table) which indicate count of all the transitions from present to the next state .The other method is CBDMRA (counter based data mining rule Algorithm) takes clock pulse, present state and next state as input from the transactional database and from these inputs, the transitions (patterns) of all the states has been found based upon its position and the time assigned to it from the transactional database. The performance analysis of both the waveforms has been generated and the proposed data mining rule algorithm proved to be more efficient and leads to minimal of time by eliminating hardware resources.**

***Keywords*—Data mining, minimum support threshold, multiple scan, Systematic Algorithm, timing algorithm.**

I. INTRODUCTION

Data mining has pulled in a great deal of attention in the information and communication technology and in the society in the recent years. It has an enormous power to turn a widely available data into useful information and knowledge. This information and knowledge, which is extracted from the raw data, can be used for many applications such as market analysis, telecommunication industry etc [2].

In this paper, data mining techniques are applied to logic circuits which highly helpful in the conversion of hardware to software implementation. Basically, Circuits for counting events are frequently used in computers and other digital systems. Since a counter circuit must remember its past states, it has to possess memory. Counters can be classified into two broad categories (i) Synchronous counters - all memory elements are simultaneously triggered by the same clock (ii) Asynchronous counters - the first flip-flop is clocked by the external clock pulse, and then each successive flip-flop is clocked by the Q or Q' output of the previous flip-flop. The introduction of flip-flops come to concern as how they are connected to determine the number of states and the sequence of the states that the counter goes through in each complete cycle[5].

A synchronous counter is a sequential machine that produces a specified count sequence. The count changes whenever the input clock is asserted. It usually consists of two parts: the memory element and the combinational element. The memory element is implemented using flip-flops while the combinational element can be implemented in a number of ways. In synchronous counters all clocks of flip-flops are connected to the same clock signal. Thus, all flip-flop outputs change state at precisely the same moment. For example consider 2-bit counter. Each state is an increment to its previous state by 1, which means that the counter is an up counter. This states that the first flip-flop (FF0) toggles at the clock edge (either rising or falling) and its input is always at high logic. The second flip-flop (FF1) toggles if the output of FF0 is 1 with its input connected to the output of the previous flip-flop[6].

The proposed method doesn't require any memory or combinational elements. It consists of pure mathematical data mining logic to implement synchronous counter logic. The main aim of this paper is to find the transitions among all the states without using any hardware components (neither memory nor combinational elements) and the clock pulse (counter) is generated automatically from the transactional database. Finding transitions among all the states is totally based on the counter and the time assigned to it from the transactional database [1].

II. CONTRIBUTIONS OF THIS PAPER

The main contributions of this work are as follows:

- A new CBDMRT (counter based data mining rule Table) has been proposed to count the number of occurrences of all the transitions of the state. Based on the count value the corresponding transitions are found using the present state and the next state. Finally transitional patterns are discovered in one scan database.
- To avoid the loss of transitions, the CBDMRA (counter based data mining rule Algorithm) has been introduced where the transitions among all the states have been generated for every transition in the transactional database and the waveforms are generated using model simulator (modelsim).

The remainder of the paper is organized as follows: Section 2 gives an overview of the basic concepts involved in synchronous counter. Section 3 deals with the concept of two proposed algorithms to discover transitions (patterns) in one scan database and also the parameters used in the algorithms. Section 4 presents Experimental results along with the performance of the proposed algorithm as how it generates the Waveforms of 2-bit synchronous counter. In Section 6 Summarizes research and discuss some future work directions.

III. RELATEDWORKS

It is a common convention that finding transitions will give us the associative knowledge of the different states present in the transactional database. This knowledge in turn is useful for finding the transitions between different numbers of states present in the transactional database.

Association rule mining is the efficient method which is used in finding the association rules. These association rules describe the associations between the attribute values of any itemsets. They can be found by means of various methods among which support and confidence [3], [4] will be considered as the optimized methods in finding them. The key to find the association rules is to find all the frequent itemsets present in the given transactional record by means of the minimum support threshold. An association rule is best expressed by means of the expression X -> Y. it means that for any occurrence of item X present in the database there is relatively high probability of occurring the item Y. here X is called as antecedent and Y is called as the consequent. The strength of such rule can only be calculated by means of its support and confidence.

A. Support

Support is the one which is used to find the strongest association rules in the itemsets. It is the most widely used measure for finding the association rules. For any given rule X => Y, support will measure total number of transactions that contains both the items X and Y in percentage.

B. Confidence

Confidence is another approach for finding the association rules. It implies the probability that any association rule will repeat itself in future transaction records. Confidence measures the number of transactional records that contains item X that also contain Y in percentage.

C. Positive CBDM Rule

The normal convention in discovering the CBDM rules is by means of finding the transitions between the present and the next state using clock pulses in the transactional database. The rules that are normally obtained by means of using minimum support threshold and minimum confidence threshold are generally referred as the positive CBDM rules and the rule is of the form A→B. That means that they are capable of associating one transition to the other transition in a given set of transactional records.

D. Negative CBDM Rules

Contrary to the positive CBDM rules described above, negative CBDM rules are defined as the rule that involves the absence of certain transitions. For example, consider A => ¬B, here '¬' indicates the absence of transitions B in a set of given transactional records. The rules are of the forms (A→ ¬B, A. Association Rules ¬A→ B and ¬A→ ¬B) are negative CBDM rules [1].

IV. ISSUES IN CONVERTING HARDWARE TO SOFTWARE IMPLEMENTATION

A. Synchronous Counter

The Features such as Internal look-ahead for fast counting, carry output for n-bit cascading, counting, Load control line, Diode-clamped inputs, Typical clock frequency 35 MHz ,Pin-for- pin replacements popular 54/74 counters are required for making synchronous counter to work. Moreover all these hardware components are much more cost effective. These components are:

1. Power Supply - This module can be built into the PLC processor module or be an external unit. Common voltage levels required by the PLC are 5Vdc, 24Vdc, 220Vac. The voltage lends are stabilized and often the PS monitors its own health.
2. Processor - This is the main computing module where ladder logic and other application programs are stored and processed.
3. Input/Output - A number of input/output modules must be provided so that the PLC can monitor the process and initiate control actions as specified in the application control programs. Depending on the size of the PLC systems the input-output subsystem can either span across several cards or even be integrated on the processor module. Some of these input-output cards generates/accept TTL level, clean signals. Output 'modules' provide necessary power to the signals. Input 'modules' converts voltage levels, cleans up RF noise and isolates it from common mode voltages. I/O modules may also prevent over voltages to reach the CPU or low level TTL.
4. Indicator lights - These indicate the status of the PLC including power on, program running, and a fault. These are essential when diagnosing problems.
5. Rack, Slot, Backplane – These physically house and connect the electronic components of a PLC.

All these hardware components are necessarily should meet the given constraints. But all these issues can be overcome by the proposed methodology.

B. Count Module

A count module senses fast pulses, from sources such as shaft angle encoders, through several input ports. Counting frequency can be as high as 2 MHz and a typically, a counter of length 16 bit or more can count up and down. Counter modules can often also be applied for time and frequency measurement and as a frequency divider.

Typical counter module hardware contains, among possible other things, an interface to the processor through the system bus, a counter electronics block, a quartz controlled frequency generator and a frequency divider. For example, it may contain, say, 5 counters with, say, 16 bits, each of which are cascadable. In this way, up to 80 bit can be counted in various codes. Thus, decimals up to about 1024 can be counted. Each port input can be switched on to the counter at random. It is possible to place a frequency divider from 1 to 16, between the port input and the counter. The frequency of an internal frequency generator can be directed either straight to a counter or via the frequency divider to a port input. On reaching the terminal count, the counter outputs a level or edge signal.

For each counter there are a number of different operating modes, which can be set by a user program. With a comparator and an alarm register, a number of count values can be compared and under defined conditions configured to turn on a process alarm [7].

A counter can be programmed in many ways, such as:

- Count mode binary or BCD coded
- Count once or cyclically
- Count on rising or falling edge
- Count up or down
- Counting of internal clock or external pulses

The only solution to solve this problem is the software implementation which works with fine environment. The one used here is the data mining applications where the synchronous counter has been framed by CBDMRT and CBDMRA which it depends completely on the algorithm to work to find transitions among all the states by giving clock pulses, present state and the next state.

V. CONCEPT OF THE PROPOSED FRAMEWORK

C. CBDMR Approach

To overcome the above said issues, we proposed a systematic algorithm to discover patterns using highest support values in the systematic table. This will take any dataset as input, and a systematic table (Table 1) is constructed for every transaction in the provided in the dataset. The systematic tables for every itemsets involved in the datasets are calculated by the following conditions [1]:

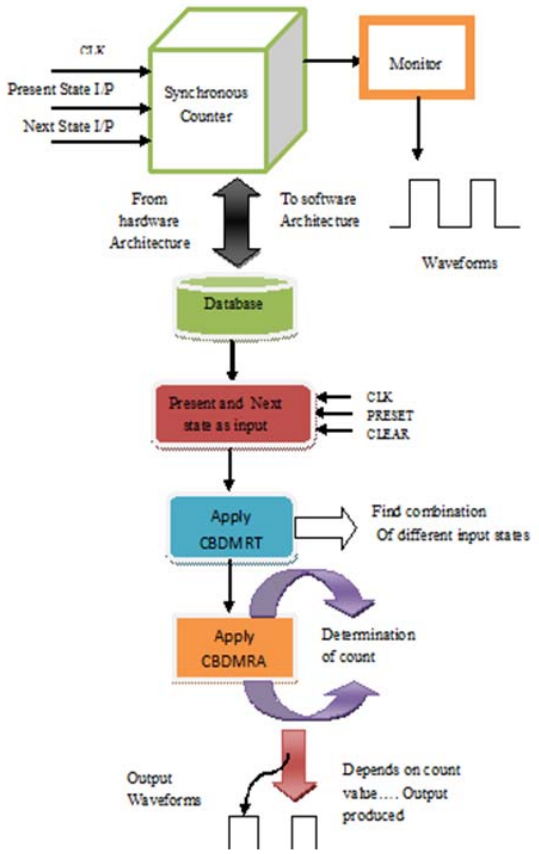

Fig 1.Proposed framework

$$\text{Supp}(A \rightarrow B) = \text{supp}(A) + \text{supp}(B) + \text{supp}(A \cup B) \quad (1)$$

$$\text{Supp}(A \rightarrow \neg B) = \text{supp}(A) - \text{supp}(A \cup B) \quad (2)$$

$$\text{Supp}(\neg A \rightarrow B) = \text{supp}(B) - \text{supp}(A \cup B) \quad (3)$$

$$\text{Supp}(\neg A \rightarrow \neg B) = 1 - \text{supp}(A) - \text{supp}(B) + \text{supp}(A \cup B) \quad (4)$$

TABLE I

PRELIMINARIES AND NOTATIONS IN THE PROPOSED FRAMEWORK

Notions	Definition
TDB	Any Transactional Database
Count(I)	Count of Pattern I in TDB
σn(I)	The n th transition of pattern I in TDB
Sn(I)	Position of pattern I in TDB
CBDMRT	Finds count of Positive and negative transitional Patterns of all states
CBDMRA	Based on transition rule the waveforms are derived as an output

Parameters Included: TDB – transactional database, CLK – clock pulse, CBDMRT – counter Based Data Mining Rule Table, CBDMRA – Counter Based Data Mining Rule Algorithm
Initialization:
Time = Assigned Time / Total Transactions
Input: Read clock pulses, present and next state input from the transactional database
Output: Discover Patterns in one scan database

Counter Based Data Mining Rule Algorithm
For each transaction in the transactional database do:
Supp (A -> B) = Supp (A) + Supp (B) + Supp (A∪B)
Supp (A -> ¬B) = Supp (A) - Supp (A∪B)
Supp (¬A -> B) = Supp (B) - Supp (A∪B)
Supp (¬A -> ¬B) = 1- Supp (A) - Supp (B) + Supp (A∪B)

Counter Based Data Mining Rule Table
L: For each transitions generated from the systematic table do
Supp (A -> B) = Supp (A) + Supp (B) + Supp (A∪B)
Supp (¬A -> ¬B) = 1 - Supp (A) - Supp (B) + Supp (A∪B)
Loop [L] until no transitions is has been left in the systematic table
Goto T.

Timing Algorithm:
T: For each of the transitions in TID do
Find the count of a pattern as
Count (I, TDB) = {(clock pulse, transition from present state to next state)}
Based on the count transitions, output waveforms are generated.

TABLE II
CBDMR TRANSITION COUNT TABLE1

		Present state	
Next state		*Y=¬A*	*¬Y=A*
	X=C	1	1
	¬X=¬C	1	1

TABLE III
CBDMR TRANSITION COUNT TABLE 2

		Present state	
Next state		*Y=¬A*	*¬Y=A*
	X=D	0	2
	¬X=¬D	2	0

TABLE IV
CBDMR TRANSITION TABLE 1

Transitions	Present states
0 →0	~A~B
0 →1	~AB
1 →0	A~B
1→1	AB

TABLE V
CBDMR TRANSITION TABLE 2

Transitions	Present states
0 →1	~A~B,A~B
1 →0	~AB,AB

VI. EXPERIMENTATIONS

The proposed Counter Based Rule algorithms, used to manifest the efficiency of the transitions from the present state to the next state along with clock pulses to show the importance of using time stamps for synchronous counter. The proposed algorithm has been applied on mod-4 synchronous counter to determine the output waveforms and this generated waveform finally matched with the waveforms generated by the hardware components of flip flop's in which the proposed algorithm proves the elimination of hardware components makes the proposed CBDMRA more effective.

D. Implementation of the Proposed Work

TABLE VI
TRANSACTIONAL DATABASE

Clock Pulses	Present and Next state	Time Stamp
1	~A ~B ~C D	25
2	~A B C ~D	50
3	A ~B C D	75
4	A B ~C ~D	100

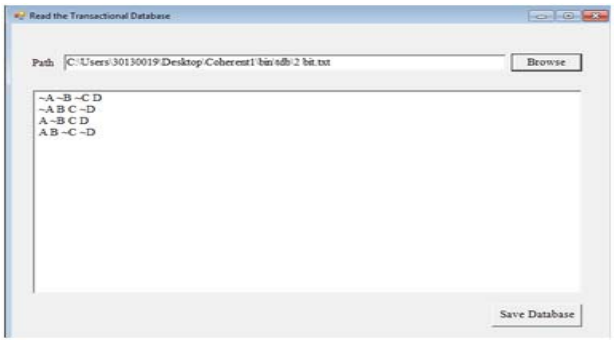

Fig 2.Reading the inputs from the transactional database

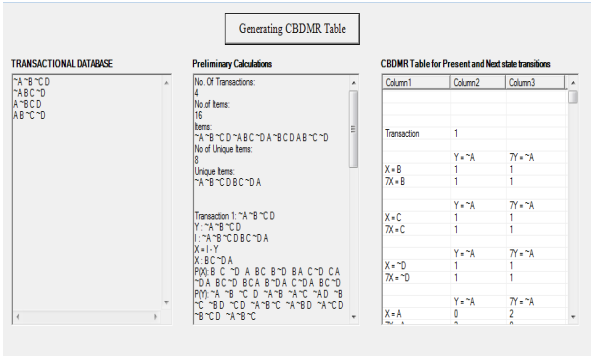

Fig 3.Generating the number of counts of all the transitions from the transactional database

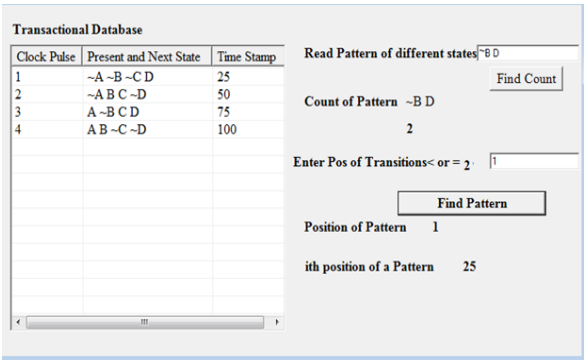

Fig 4.Finding the number of counts and position of all the transitions from the transactional database.

VII. COMPARISON AND PERFORMANCE ANALYSIS

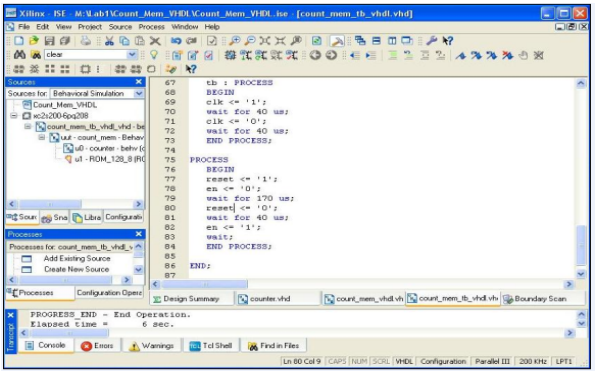

Fig 5.Verilog coding for mod4 Synchronous counter

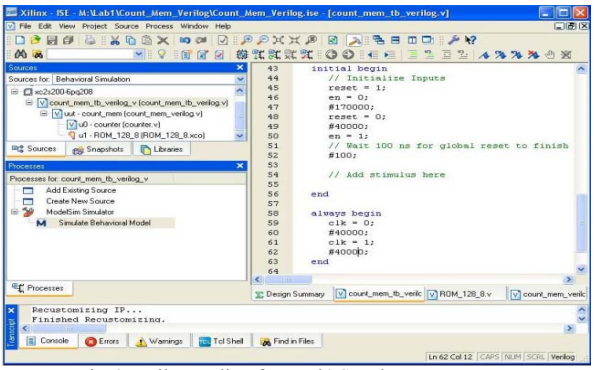

Fig 6.Verilog coding for mod4 Synchronous counter

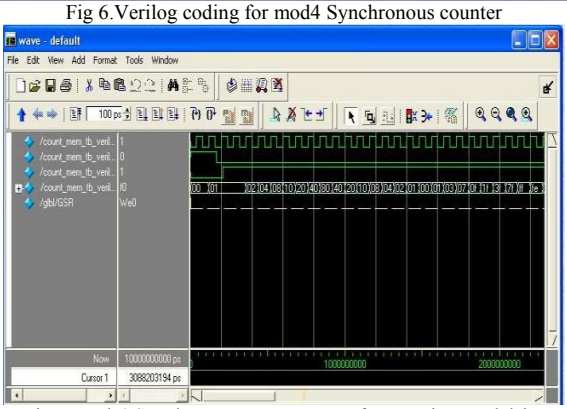

Fig 7.Mod-4 Synchronous counter waveforms using modelsim

The hardware implementation of the synchronous counter is time consuming process, requires number of hardware components which leads to complication. These issues can overcome by the proposed methodology by eliminating hardware components.

TABLE VII
CBDMRA TABLE FROM TRANSACTIONAL DATABASE

Clock Pulses	Present and Next Sate as Input
1	~A ~B ~C D
2	~A B C ~ D
3	A ~B C D
4	A B ~C ~ D

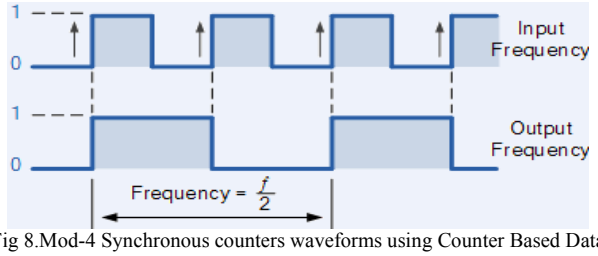

Fig 8.Mod-4 Synchronous counters waveforms using Counter Based Data Mining Rule Algorithm

VIII. CONCLUSION

The proposed algorithm, Counter Based Data Mining Rule Algorithm (CBDMRA) will be very efficient in finding the transition rules among different states from the transactional database. The use of time stamp also helps in avoiding the multiple scans across the database. This will leads to better performance (search space), minimization of time and useful in effective decision makings.

REFERENCES

[1] S. Sangeetha, "Verdict of Association Rule Using Systematic Approach of Time Slicing for Efficient Pattern Discovery" IEEE Digital Xplore in 2012.

[2] R.Agrawal, T. Imielinski, and A. Swami, “Mining Association Rules between Sets of Items in Large Databases,” Proc. 1993 ACM SIGMOD

[3] R.Agrawal and R.Srikant, “Fast Algorithms for Mining Association Rules,” Proc. 20th Int’l Conf. Very Large Data Bases, pp. 487-499, 1994.

[4] R.Srikant and R.Agrawal, “Mining Generalized Association Rules,” Future Generation Computer Systems, vol. 13, nos. 2/3, pp. 161-180, 1997.

[5] W. Aloisi and R.Mita, “Gated-Clock Design of Linear- Feedback Shift Registers,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol.55, no.6, pp.546–550, June 2008.

[6] V.Stojanovicetal. “Comparative analysis of master–slave latches and flip–flops for high-performance and low-power systems,” IEEE J. Solid-State Circuits, vol.34, pp.536–548,April. 1999.

[7] Jan M. Rabaey, Digital Integrated Circuits, PHI LEARNING Private Limited. Edition – 2003.

[8] G. Yee and C. Sechen, “Clock-delayed domino for adder and combinational logic design" in proc.IEEE/ACM Int. Conf. Computer Design, Oct., 1996, pp. 332-337.