Abstract—Leakage power is the main dominant source of power dissipation for Sub-100nm VLSI circuits. Various techniques were proposed to reduce the leakage power dissipation; one of these techniques is Multi-Threshold voltage. In this paper, the exact and optimal values of Threshold Voltage ($V_{th}$) for each transistor of the design are found for any sequential circuit. This is achieved by applying Artificial Intelligence (AI) Search Algorithm to find the optimum values that get the largest reduction of leakage current. The proposed algorithm exploits the total Slack Time of each transistor’s location and their contribution of leakage current. The proposed algorithm is introduced by AI Heuristic Search, under 22nm BSIM4 foundries predictive model. The proposed approach saves around 80% of the Sub-threshold Leakage current without degrading the performance of the circuit.

Keywords- Artificial Intelligence, Leakage Current, Low Power, Multi-Threshold Technique, Nanotechnology, SPICE Parameters.

I. INTRODUCTION.

The growing demand of portable and wireless electronic equipment at the nanotechnology era makes the saving power to be the most important issue in current VLSI circuits and systems [1]. The power dissipation in CMOS digital circuits consists from dynamic and static power dissipation. The dynamic power dissipation depends quadratically on the supply voltage $V_{dd}[2]$. Thus, reducing the supply voltage is the most effective way to reduce the dynamic power dissipation. But, the reduction of the supply voltage $V_{dd}$ increases propagation delay. But this increment violates the critical time. To overcome this problem, the threshold voltage $V_{th}$ should be reduced by the same factor of reducing the supply voltage in order to maintain the circuit frequency without change. The process of reducing $V_{th}$ value is performed by reducing the gate oxide thickness $T_{ox}$. Unfortunately, reducing $V_{th}$ in $\eta$-technology causes an exponential increment at the Sub-threshold and Gate Leakage currents [3].

Several techniques were proposed to reduce the Sub-threshold Leakage power. In [2-4], they take the advantage of process option by using the Low-$V_{th}$ for the transistors in the critical path, and High-$V_{th}$ for the transistors out of the critical path to minimize leakage power. This option is given to the designers to use transistors that are either high threshold voltage (slow but low leakage) or low threshold voltage (fast but high leakage) by only an extra mask layer [5].

In [3], they proposed a simple Breadth First Search Algorithm to find the gates in the design that should have High-$V_{th}$, and which have Low-$V_{th}$. Unfortunately, the search is applied at the gate level, not the transistor level. In [6], they used Integer Linier Programming (ILP), the results show more saving power than [3], but it takes more delay time, even though it is not applicable for large circuits. Another approach, which uses Depth First Search was proposed in [4], it gives the critical path gates Low-$V_{th}$ and High-$V_{th}$ for the gates out the critical path.

From this introduction, the references [3,4,6] discussed the leakage current reduction from gate level of view, and using only two fixed values of threshold voltages (high and low). From our view, those approaches are not so intelligent to reach the optimal reduction of the leakage current. An efficient technique for reducing the leakage current for $\eta$-scale CMOS circuits is the Multi-Threshold voltage technique. It is based on using various threshold voltages in the circuit to reduce the leakage current at minimal value. The high values of threshold voltage are assigned to the transistors in the non-critical path, while the transistors in the critical path still have low threshold voltage to keep circuit frequency unchanged. This technique is considered as the solution of scaling down the supply voltage and $V_{th}$ at the same time. Multi-Threshold voltage technique is better than Dual-Threshold voltage technique, this is because it achieves more reduction of leakage power, and more exploitation of circuit Slack Time. Since the leakage current in CMOS at $\eta$-technology is considered as the main problem of power dissipation, Multi-Threshold voltage technique and Artificial Intelligence (AI) could be used together to overcome this problem. AI Search is a widely used technique to find the optimal solution for many problems, there are many AI algorithms that can be applied to solve VLSI problems, such as Search Algorithms, Genetic Algorithm, Neural Networks, etc. In this paper, AI Search is implemented to find the optimal reduction of leakage power for CMOS sequential circuits.

Section II provides the background and preliminaries. Section III describes the algorithm, which is called LOAIS. Results and discussion are shown at section IV. Section V and VI discuss the scalability and the comparison results of the proposed algorithm. The Conclusion is given at the end of paper in section VII.

II. BACKGROUND AND PRELIMINARIES.

A. Leakage Power Estimation.

The dominant contribution of power in $\eta$-technology VLSI circuits is the leakage power. It depends largely on three types of leakage currents [6], these currents are: Sub-threshold Leakage Current, Gate Leakage Current and Reverses-Biased Leakage Current (BTBT). Sub-threshold Leakage current is the main dominant source of power dissipation. It increases when the technology is scaled down below 90 nm because of threshold voltage reduction. Sub-threshold Leakage power mainly depends exponentially on both gate-to-source voltage ($V_{gs}$) and threshold voltage ($V_{th}$). In general, the formulation of Sub-threshold Leakage Current is explained at [7, 8]. Sub-threshold Leakage power also depends on the input vector and whether the transistor is turned On or Off. It occurs when the transistor is turned Off ($V_{gs} < V_{th}$).
B. Circuit Timing and Delay Estimation.

In this paper, circuit timing parameters and delay time are calculated and estimated accurately depending on RC model. In CMOS circuits, each transistor has timing parameters, which can be estimated by performing static timing analysis simulation. However, these timing parameters are Arrival Time, Required Time, Slack Time, and the circuit critical time (circuit clock). The Slack Time of a transistor is the difference time when the signal reaches the transistor and when the signal is produced by the transistor [9]. The Slack Time is very important because the proposed algorithm uses it as a guidance value to reach the optimal reduction of leakage power. This is happened when assigning transistors with High-Vth in the circuit.

III. PROPOSED ALGORITHM

The most influential technology parameters on \( V_{th} \) are the oxide layer thickness (\( T_{ox} \)) and channel doping concentration (\( N_{DEP} \))[8]. In this paper, the modification of \( T_{ox} \) is adopted to increase the threshold voltage value \( V_{th} \) for some transistors in order to suppress the leakage current. Therefore, the influence of increasing the threshold voltage for any transistor is an increment at the delay of that transistor. Thus, the proposed idea is to increase the \( T_{ox} \) for the transistors that are located out of the critical path to suppress their leakage without affecting the circuit time. The \( T_{ox} \) values start from 1.8 \( \mu m \), which is provided by BPTM, to 2.8 \( \mu m \), which is the maximum value of \( T_{ox} \).

This range can be divided to different levels.

A. Extracting Threshold Voltage Values Under 22nm Technology.

In order to estimate the leakage power, \( V_{th} \) must be computed for every \( T_{ox} \) in the design because the leakage current is related directly to \( T_{ox} \). The equation of \( V_{th} \) is used in simulation to calculate the values of \( V_{th} \) based on the change at transistors oxide layer thickness. In this paper, different levels of \( V_{th} \) are used in the algorithm, the lowest level (level 0) at \( V_{th} = 0.25V \) and \( T_{ox} = 18 \ A \), and the highest level (level 10) at \( V_{th} = 0.4V \) (\( T_{ox} = 28 \ A \)). The values of \( V_{th} \) for these levels were calculated by substituting \( T_{ox} \) in its equation [8].

B. Multi-\( V_{th} \) Assignment Algorithm

In order to save leakage power, Multi-\( V_{th} \) technique is applied by assigning the transistors, which do not affect the circuit critical time, with High-\( V_{th} \) by increasing their \( T_{ox} \). The algorithm exploits the Slack Time for each transistor by increasing its \( T_{ox} \) with acceptable increase at its delay without affecting the circuit frequency. The algorithm is called Leakage Optimization using Artificial Intelligence Search (LOAIS), and consists of three stages: Initialization, Static Timing Analysis, A* Search for Multi-\( V_{th} \), the details of the above steps are shown below:

1) Initialization.

Any circuit is transformed from its state to a graph state. This graph is a Cyclic Graph type because it deals with sequential circuits. As a start, the circuit is transformed to a graph \( G (V, E) \) that is composed from vertices, which are transistors in the circuit, and edges, which are the connections between transistors vertices.

2) Static Timing Analysis.

This step is responsible for estimating the timing analysis for the proposed circuit, these timings depend mainly on the estimation of transistors propagation delays. The delay estimation process is based on the RC model. After finding the delay propagation of transistors, other timing parameters like Arrival, Required, Slack, setup, C-Q, clock are calculated and should be ready to be used in the next stage[9][10]. The timing analysis is produced based on the minimum value of \( T_{ox} \) for the used technology.

3) A* Search for Multi-\( V_{th} \)

This step is responsible for applying A* Search Algorithm to find the optimum values of \( V_{th} \). Later on, each value of these thresholds will be assigned for a specific transistor in order to reduce the leakage current at the lowest value.

Slack Time represents the amount of time that could be used to delay the transistors output from its original delay without violating the circuit critical time and frequency. Our proposed algorithm depends mainly on the Slack Time. One of the ways that could be used to increase transistors delays is by increasing its \( T_{ox} \) value, so the increase of delay can be exploited to minimize the leakage current and power. The Slack Time of a transistor depends on the circuit paths where the transistor is involved in. Therefore, any change at any transistor Slack Time will affect the Slack Time of the other transistors placed in the same path with that transistor. Any increment at \( T_{ox} \) level for a certain transistor will reduce its leakage power and this leads to reduce the average leakage power for the whole circuit.

That’s because every transistor in the circuit contributes in the Sub-threshold leakage power. This contribution depends on transistor size, transistor \( V_{th} \), and circuit inputs. Where circuit inputs determine which transistors are turned On and which are turned Off.

The A* Search is Heuristic Search that uses two functions to reach the goal state. The goal state for any sequential circuit is the state which has the minimum leakage power and maximum total Slack Time. The total Slack Time of a circuit is the summation of all slacks for the circuit transistors. The A* Algorithm searches for the most reduction of leakage power for any sequential circuit by applying Multi-\( V_{th} \) technique with maintaining the circuit frequency without changing. Actually, some experiments show a speed up in the circuit frequency. The A* Algorithm is applied on the flip-flops transistors before the combinational sub-circuit transistors to get the most reduction of leakage power.

Fig. 1 shows the LOAIS Algorithm. The algorithm is initialized first to copy the circuit clock and leakage. Then the circuit is transformed to the cyclic graph \( G (V, E) \), followed by calculating the circuit timing parameters at L01. First, the circuit state is enquired inside the Queue and then queued at L02 and L04 after checking the content of Queue, that’s because there is Go To statement at the end of the algorithm which returns to L03. At L05 and L06, circuit transistors which are located out of the critical path are chosen one by one. For each chosen transistor, oxide thickness is increased, as shown at L07 and L08. Then the circuit leakage and frequency is calculated after modification of that transistor. If the circuit has less leakage and the frequency doesn’t increase, then the circuit state will be accepted and inserted into the Queue at L10. At L11 and L12, the circuit returns to its previous state, and then the algorithm chooses another transistor to modify its \( T_{ox} \). If the circuit frequency increases or the circuit leakage doesn’t decrease, then this state will be deleted from memory, as shown at L15. Finally, the best state, which was saved in the Queue, is extracted to modify other transistors oxide thickness again as L04. This is performed to try all combinations of modification with a guided sequence from A* Algorithm.

Fig. 2 shows an example of the states, which are generated from A* Algorithm. At each depth of search, there are many generated states, these states are expanded by increasing \( T_{ox} \) for circuit transistors. The new states are inserted in the Queue or the circuit leakage doesn’t decrease, then this state will be deleted from memory, as shown at L15. Finally, the best state, which was saved in the Queue, is extracted to modify other transistors oxide thickness again as L04. This is performed to try all combinations of modification with a guided sequence from A* Algorithm.
is increased; also some states are deleted because they don’t lead to
the goal.

IV. SIMULATION RESULTS AND DISCUSSION.

The implementation of the proposed algorithm is performed
using own-built simulator, it is designed using C++ programming
language. It incorporates BSIM equations to describe the behavior of
MOS transistor at 22gm technology. The simulations are run at room
temperature of 25°C, the leakage current in this case leaks from the
circuit when the circuit is in sleep mode. The transistors’ channel length
and decrease transistors mobility was considered when the transistor channel width
was determined.

Two benchmark circuits were tested to find the optimum reduction of Sub-threshold
Leakage and the optimum value of High-Vth. These circuits are S27 bench mark from ISCAS’89, and
and B02 from ISCAS’99. The output of the simulation of LOAIS Algorithm with Multi-Vth technique for both circuits are shown in
Table I. The table shows the levels used for Vth and the number of transistors which have High or Low Vth. Also from the table, the
results of the estimated leakage power before and after applying Multi-Vth technique, and the saving of leakage power are shown too.
Table I shows the critical time (clock) of S27 and B01 circuits before and after applying the technique, it’s obviously that the clock is either
remains unchanged or speed up with 0.033%, thus this is another benefit from LOAIS Algorithm.

This improvement in speed is produced from the change of gate capacitance for some transistors, which is caused by Vth going
thinner. In LOAIS Algorithm, it is impossible to violate circuit critical time because such states are ignored from the Search
Algorithm. From Table I, the optimum state of the circuit is by using Multi-Vth technique with 4 levels for S27 circuit to get a reduction of
leakage power reaches to 70.77% with a speed up in circuit clock

Figure 1: LOAIS Algorithm for Multi-Vth technique.

Some states at the same level of search are inserted into the Queue, as shown before in the algorithm, because they have the same
Slack Time. In fig. 2, the vertical lines nodes can lead to a solution in
the A* Search, they are inserted in the priority Queue, and could be
deleted later because they may not lead to an optimal state. On the
other hand, the white nodes are already deleted from the moment of
generation because they have less Slack Time than others. The states
that increase circuit frequency have been already deleted from the
beginning of generation, so these states are not shown in the figure.

V. A* LOAIS Algorithm.

The computational requirements for most interesting search problems grow very quickly with problem size, so any search
algorithm has a problem with scalability. In A* Search, the scalability is
faced by pruning the unnecessary states, which don’t lead to the
goal state. In general, scalability problem can be easily solved by
using high performance hardware or using parallel computation.

The goal state achieves the most reduction of Sub-threshold Leakage power. The branching factor for the A* Search equals the
number of circuit transistors. So, if the circuit has 1000 transistors, the number of generated states at each level equals 1000 too. Table II
shows the relation between the number of generated states in the search space and the number of circuits transistors for each level of
search. From the table, it is clear that when the number of transistors is increased, the number of generated states is increased too. Also for
each circuit, the number of generated states is increased with each increase at the number of levels. It also shows the relation between
the depth of goal state in search space and the number of the
transistors for each level of search. It is clear that when the number of circuit transistors is increased, then the depth of goal state is
increased too. Also for each simulated circuit, the depth of goal state is increased with each increase at the number of levels.

VI. COMPARISON RESULTS.

Fig.5 shows the number of generated states for the simulated
circuits, when they implemented by [3] and LOAIS. The result is that
the number of generated states in LOAIS is much less than the
generated states by [3]. That’s because they used Breadth First
Search which is not Heuristic Search. The results of Leakage Power
reduction by [3] and LOAIS is the same because both of them are complete in terms of Artificial Intelligence Search.

Fig. 6 shows the reduction of leakage power for the simulated
circuits. It is clear that LOAIS Algorithm achieves more reduction of
leakage, but this achievement is not so large compared to [3]. This is
because we have implemented the algorithm of [3] at transistor level
rather than gate level, which is the main issue of reduction for CMOS
circuits, because there are many transistors inside the critical path
gates can be altered to reduce their leakage current.
In this paper, we have proposed LOAIS Algorithm to assign Multi-$V_{th}$ to a maximum number of transistors for any sequential circuit at transistor level. The algorithm could be applied to get the lowest value of Sub-threshold Leakage power without affecting the circuit frequency. The proposed algorithm doesn’t need high memory resources because it ignores any deleted state in the search. However, experimental results shows the reduction of leakage which reaches 78% with more speed up of circuit clock.

VII. CONCLUSION.

In this paper, we have proposed LOAIS Algorithm to assign Multi-$V_{th}$ to a maximum number of transistors for any sequential circuit at transistor level. The algorithm could be applied to get the lowest value of Sub-threshold Leakage power without affecting the circuit frequency. The proposed algorithm doesn’t need high memory resources because it ignores any deleted state in the search. However, experimental results show the reduction of leakage which reaches 78% with more speed up of circuit clock.

**REFERENCES**


---

**Table I:** Leakage power search Algorithm output for S27 and B02 Benchmarks circuit.

<table>
<thead>
<tr>
<th>Bench Mark</th>
<th>$T_{on}$ levels</th>
<th># of transistor have</th>
<th>Estimated leakage before e-7A</th>
<th>Estimated leakage after e-7A</th>
<th>Saving leakage power</th>
<th>Critical time before e-11sec</th>
<th>Critical time after e-11sec</th>
<th>Speed up</th>
<th>Slack</th>
</tr>
</thead>
<tbody>
<tr>
<td>B02</td>
<td>0.7</td>
<td>25</td>
<td>9.683</td>
<td>3.306</td>
<td>65.86%</td>
<td>3.01020</td>
<td>3.00946</td>
<td>0.025%</td>
<td>47.2%</td>
</tr>
<tr>
<td>S27</td>
<td>0.5,10</td>
<td>24</td>
<td>9.683</td>
<td>3.011</td>
<td>68.90%</td>
<td>3.01020</td>
<td>3.01010</td>
<td>0.028%</td>
<td>57.25%</td>
</tr>
<tr>
<td></td>
<td>0.2,6,10</td>
<td>29</td>
<td>9.683</td>
<td>2.830</td>
<td>70.77%</td>
<td>3.01020</td>
<td>3.00922</td>
<td>0.033%</td>
<td>57.80%</td>
</tr>
<tr>
<td></td>
<td>0,2,3,5,8</td>
<td>31</td>
<td>9.683</td>
<td>2.993</td>
<td>69.09%</td>
<td>3.01020</td>
<td>3.00942</td>
<td>0.026%</td>
<td>50.85%</td>
</tr>
</tbody>
</table>

**Table II:** The number of generated states and the depth of goal state values for the simulated sequential circuits.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>No. of generated states</th>
<th>Goal Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>S27</td>
<td>770</td>
<td>245</td>
</tr>
<tr>
<td>B02</td>
<td>1200</td>
<td>3384</td>
</tr>
</tbody>
</table>

**Figure 5:** Comparison between LOAIS Algorithm and Dual-$V_{th}$ Algorithm in [3] according to the number of generated states.

**Figure 6:** Comparison between LOAIS Algorithm and Dual-$V_{th}$ Algorithm in [3] according to the reduction of $I_{leak}$. 