Power Invariant Vector Compaction based on Bit Clustering and Temporal Partitioning

Nicola Dragone nicola.dragone@st.com
Roberto Zafalon roberto.zafalon@st.com
Carlo Guardiani carlo.guardiani@st.com
Cristina Silvano* silvano@bsing.ing.unibs.it

STMicroelectronics, Central R&D, I-20041 Agrate B. (MI), ITALY; Tel: + 39 39 603 5313
*Università degli Studi di Brescia, DEA, I-25123 Brescia, ITALY; Tel: + 39 30 3715 445

1. ABSTRACT

Power dissipation in digital circuits is strongly pattern dependent. Thus, to derive accurate simulation-based power estimates, a large amount of input vectors is usually required. This paper proposes a vector compaction technique aiming at providing accurate power figures in a shorter simulation time for complex sequential circuits characterized by some hundreds of inputs. From pairwise spatio-temporal signal correlations, the proposed approach is based on bit clustering and temporal partitioning of the input stream aiming at preserving the statistical properties of the original stream and maintaining the typical switching behavior of the circuit. The effectiveness of the proposed approach has been demonstrated over a significant set of industrial case studies implemented in CMOS submicron technology. While achieving a 10x to 50x stream size reduction, the reported results show an average and maximum errors of 2.4% and 7.1% respectively, over the simulation-based power estimates derived from the original input stream.

1.1 Keywords

Power Estimation, Vector Compaction, Markov Chains, Low Power VLSI Design.

2. Introduction

Power estimation techniques can be classified into two classes: simulative and probabilistic techniques [1], [2], [3], [4]. The first one provides good accuracy at the expense of high computational cost. The second approach is generally faster but less accurate. The level of accuracy required during critical design steps, like the final circuit verification sign-off or the sizing of chip package and heat sink, could require electrical level simulations. However, given the complexity of existing VLSI blocks, most of the low level simulation engines will land in overwhelming difficulties and unacceptable computational effort. In these cases, automatic techniques for size reduction of the input stream can play a primary role to shorten the simulation time, while maintaining the main power behavior of the circuit. The compaction problem we want to address can be formally stated as: *"Compact a sequence of input vectors S1 of length L1, into a new sequence S2 of length L2 << L1, suitable to preserve the activity of the internal and I/O nodes of the circuit and the overall average power dissipation. The compaction ratio is defined as the factor **R = L1 / L2** ".*

An effective power invariant stream compaction algorithm does not need to exactly reproduce the same logic behavior as that caused by the original stream. The compacted sequence should only move the internal nodes such that the average power consumption is nearly the same. Furthermore it's worth noting that the average power derives from the contribution of many different functional modes of the circuit. Therefore it is important to assure that the compacted sequence holds the same fraction of each of these modes.

The proposed power invariant compaction algorithm is based on bit clustering of the input word and does not require any direct knowledge of the internal structure of the circuit. Basically, bit slices belonging to different clusters are compacted independently. By automatically tracing the most important spatio-temporal correlations of the input vectors, the compacted sequence has an high probability of sensitizing the internal nodes of the circuit with the same switching activity as the original sequence. The major difference between our technique and other existing compaction procedures is the capability of handling circuits with a large set of inputs. In addition, an effective heuristics for the temporal partitioning of the initial sequence into subsequences has been derived, in order to capture and correctly reproduce complex timing dependencies of the original sequence.

3. Vector Compaction Technique based on Bit Clustering and Temporal Partitioning

The majority of the long simulation sequences associated to complex industrial circuits with hundreds of I/Os are characterized by a heterogeneous input behavior, in both the spatial and temporal dimensions. Typically, the spatial behavior reflects the type of input bit (control bit, data bit, etc.), while the temporal behavior reflects the different operating modes. The proposed compaction method exploits these features by means of input word bit clustering and temporal partitioning of the original simulation sequence.

3.1 Bit Clustering

The purpose of bit clustering is to group together bits of the original input vectors according to their correlation value: bits belonging to the same cluster are maximally correlated, whereas bits of different clusters present very low correlation values. As a matter of fact, it is common to find input streams in which control bits are highly correlated among

ISLPED98, Monterey, CA, USA
© 1998 ACM 1-58113-059-7/98/0008..$5.00

themselves and far less correlated with data bits, or even input streams in which only few control bits toggle in correlation with some data and address bits. In our method, we suggest to sacrifice the lower order correlations among clusters, providing a negligible impact on power consumption, and to consider only higher order spatio-temporal correlations within clusters of correlated bits, which dramatically impact the power consumption. The proposed approach attempts to preserve this fundamental statistical information in very compacted sequences.

We can represent the bit signals like vectors in a multi-dimensional space. The problem of bit clustering consists of identifying the most highly correlated bit vectors and grouping them into clusters, which is equivalent to clustering together the closest bit vectors. We assume to measure spatio-temporal correlations by using a pairwise approximation and we adopt a symmetric encoding of their value. Assuming the hypothesis of stationarity we can express the correlation coefficient between the bits b_i and b_j as:

$$\rho_{i,j}(\tau) = \frac{Cov(b_i, b_j, \tau)}{\sigma_i(\tau, L-1)\sigma_j(0, L-\tau-1)} \tag{1}$$

Where Cov is the covariance at time τ between b_i and b_j , σ is their standard deviation and L is the stream length. Since the correlation among bit pairs is not only *spatial,* but also *spatio-temporal*, we need to consider non zero values for τ. Setting the maximum value for τ equal to τ_{max}, we can collect the pairwise correlation coefficients in (τ_{max}+1) correlation matrices $C(\tau)$. The value of τ_{max} is theoretically the maximum *correlation length* in the input stream. In our experiments, we assumed τ_{max} equal to 10, showing a sufficient accuracy in capturing correlations, while maintaining a limited memory requirement and still acceptable computational time. Once the correlation matrices are computed, we can apply the bit clustering algorithm. The first step of the algorithm consists of the computation of the maximum of the correlation coefficients of each pair of bits. Then an iterative procedure performs the bit clustering based on the maximum correlation criteria applied to each pair of bits. A final refinement, consisting of merging the residual small clusters together, is performed at the end.

3.2 Temporal Partitioning

In our compaction procedure we introduced some heuristics for partitioning the initial sequence into consecutive blocks, trying to identify different activity behaviors of each bit. In particular, we define three *bit dynamics*: stable high, stable low and toggling. A bit has a stable high (low) dynamic when a high (low) logic value is asserted for at least T consecutive time steps, where T is a user defined parameter. A bit has a toggling dynamic when it has neither a stable high nor a stable low dynamic. An application of these definitions is shown in figure 1. The proposed partitioning algorithm tries to identify among all the discontinuous bit points a set of significant *partitioning break-points.*

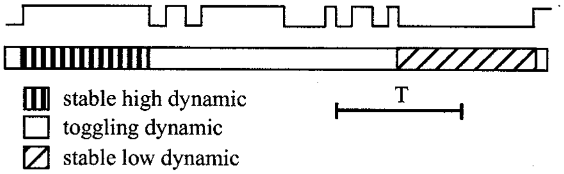

Fig. 1:Different bit dynamics

In figure 2 we show the pseudo code of the partitioning algorithm aiming at capturing a common dynamic behavior in the sequence, starting from the different dynamic behavior of each bit vector b_i . A final refinement is adopted to split large blocks where no change in bit dynamics occurs. Clearly, the partitioning results tightly depend on the value of T, determined by heuristics. By temporal partitioning, first we reduce the amount of vectors to be processed on the temporal block. Second, we extract blocks in which the probability of having one or more constant bits is very high and then avoiding their inclusion in the *Dynamic Markov Tree* (DMT) built during the compaction procedure. Third and most important, the method allows to preserve discontinuities in the bit dynamics (*long correlations)*, otherwise difficult to be captured by using *Dynamic Markov Models* (DMMs) [5]. These long correlations are often related to control bits (e.g. chip select or output enable) that are crucial to preserve the global power figure.

```
for each bit $b_i$ (i=0,...,N-1)
  trace dynamics $D_i$ of $b_i$;
  for each toggling dynamic $T_i$ of $b_i$
    if (size($T_i$) < $size_{min}$) merge $T_i$ with the shortest
      adjacent stable dynamic;
Set list of sequence partitioning points $L_{pp}$ = empty;
for each bit $b_i$
  for each dynamics $D_i$ of bit $b_i$
    if (for each element j in $L_{pp}$ | $d_{i,start}$ - $L_{pp}[j]$ | > $size_{min}$)
      add $d_{i,start}$ in $L_{pp}$;
    if (for each element j in $L_{pp}$ | $d_{i,stop}$ - $L_{pp}[j]$ | > $size_{min}$)
      add $d_{i,stop}$ in $L_{pp}$;
    (where $d_{i,start}$ and $d_{i,stop}$ are the temporal boundaries of $D_i$)

final partitioning refinement;
```

Fig. 2:Pseudo code of the temporal partitioning algorithm.

3.3 Compaction Procedure.

The compaction procedure is applied to each bit cluster by using the DMMs [6]. To improve the effectiveness of this technique, we exploited the possibility to operate separately on each cluster using different lengths for the DMT. The computation of the actual length is based on the above mentioned concept of correlation length: from the (τ_{max}+1) correlation matrixes, used during bit clustering, we also obtain the maximum correlation length, i. e. the DMT length for each cluster. The pseudo code of the algorithm, shown in figure 3, is based on checking the correlation values against a threshold, that is considered as an increasing function of τ.

During the compaction procedure, non-original vectors are

TABLE 1: STREAMZIP RESULTS

Circuit	# of Inputs	Circuit Complexity	Original Stream		Compact. Ratio	Avg Power Compacted Stream	% Error Avg Power	CPU Time		
			# of Vectors	Average Power				Seq. Part..	Bit Cluster.	Compact.
24-bit Multiplier	24 bit	8,000 gates	2002	6.9 mW	10	6.9 mW	0 %	0.11 s	8.43 s	0.41 s
Bus Sequencer	33 bit	2,000 gates	10000	10.8487 mW	20	10.8495 mW	0.007%	0.27 s	81.26 s	4.68 s
SRAM	25 bit	5,000 trans.	4403	1.214 mW	20	1.270 mW	4.6%	0.13 s	20 s	2.02 s
ST7 8-bit μC	103 bit	4,500 gates	1497	448 uW	10	416 uW	7.1%	0.15 s	115.39 s	2.59 s
M8051 8-bit μC	53 bit	7,000 gates	60000	20.9617 mW	50	20.6663 mW	1.4%	2.1 s	1303.07 s	37.01 s
ST20 32-bit RISC μP	141 bit	300,000 gates	75000	3.8399 mW	10	3.8999 mW	1.6 %	3.38 s	6125.8 s	125.6s
DTW* Speech Recognizer	75 bit	70,000 gates	10807	1559.9 uW @ 3.3V 319.28 uW @ 2.0V 72.28 uW @ 1.0V	10	1571.04 uW 325.25 uW 75.314 uW	0.7 % 1.9 % 4.2 %	0.52 s	422.44 s	13.05 s
AVERAGE					10 to 50		2.4 %			

*DTW measured on real silicon through a test machine

in general allowed, but some constraints can be imposed to avoid the usage of the forbidden vectors.

```
for each cluster g
    $dmtLength_g = 0$;
    for each pair of bits $b_i, b_j$ in g (also for i = = j)
        for $\tau = 1,...,\tau_{max}$
            if ($\rho_{i,j}(\tau) > c_{threshold(\tau)}$)
                $dmtLength_g = \tau$;
```

Fig. 3:Pseudo code for the computation of DMT length.

4. Experimental Results

All of the design benchmarks but the SRAM (see Table 1) have been synthesized on a general purpose CMOS standard-cell library, featuring 0.35 μm minimum gate length, 5 metal interconnection layers. The presented technique has been implemented in a tool named "**StreamZip**", that has been thoroughly tested and applied to the above cited design cases. The results reported in Table I have been obtained with a compaction ratio **R** ranging from 10 to 50. They show an average and maximum errors of 2.4% and 7.1% respectively, thus demonstrating the effectiveness of the proposed technique. In particular, it is worth noticing the extremely low error introduced when processing the first two test cases, that is mainly due to the pure combinational nature of the multiplier and the shallow sequential nature of the Address Bus Sequencer, which helps the spatio-temporal compaction algorithm. On the other hand, the largest error (7.1%) is related to a 8-bit complex instruction set microcontroller, executing a control-intensive, relatively short test pattern (less than 1500 vectors in the original stream). Bit clustering happened to be the most CPU intensive task, also due to the high τ_{max} (*max correlation length)* used in the experiments.

5. Conclusions

An innovative methodology for the power invariant compaction of input streams has been presented in this paper. The proposed methodology improves the existing techniques by introducing bit clustering and temporal partitioning of the original sequence. By capturing the operating modes performed by the circuit according to the original stream, we considerably enhance the accuracy of results, particularly for large sequential blocks that are typical of real-life industrial designs. The effectiveness of the proposed technique has been thoroughly demonstrated on a number of industrial strength test cases, achieving high compaction ratios without a remarkable loss of accuracy in the average power estimates. Furthermore the method has been also applied to a test stream for a real chip, showing a very good matching between original and compacted sequence average power figure, as measured on working silicon.

6. References

[1] F.N. Najim, "A Monte Carlo Approach for Power Estimation", IEEE, Transactions on VLSI Systems, Vol.1, No.1, pp.63-71, Mar. 1993.

[2] A. Gosh, S. Denvadas, K. Keutzer, and J. White, "Estimation of Average Switching Activity in Combinatinal and Sequential Circuits", in Proc. ACM/IEEE Design Automation Conference, pp. 253-259, June 1992.

[3] F.N. Najim, J. N. Kozhaya, "Accurate Power Estimation for Large Sequential Circuits", ICCAD-97, S. Jose' (CA), November 1997.

[4] E. Macii, M. Pedram, F. Somenzi, "High-Level Power Modeling, Estimation, and Optimization", Proc. ACM/IEEE Design Automation Conference, June 1997.

[5] G.V. Cormack, R.N. Horpool, "Data Compression Using Dynamic Markov Modeling", Computer Journal, Vol 30, n. 6, 1997.

[6] D. Marculescu, R. Marculescu, and M. Pedram, "Hierarchical Sequence Compaction for Power Estimation", in Proc. ACM/IEEE Design Automation Conference, June 1997.