TRANSISTOR SIZING OF LCD DRIVER CIRCUIT FOR TECHNOLOGY MIGRATION

Takahito Ijichi[†], *Masanori Hashimoto*[†], *Shingo Takahashi*[‡], *Shuji Tsukiyama*[‡] *and Isao Shirakawa*[††]

Dept. ISE, Osaka Univ.[†] Dept. EECE, Chuo Univ.[‡] Dept. PMI, Univ. of Hyogo[††]

ABSTRACT

Design automation of LCD driver circuits is not sophisticatedly established. Display fineness of an LCD panel depends on a performance metric, ratio of pixel voltage to video voltage (RPV). However, there are several other important metrics, such as power consumption and area, and the best circuit cannot be decided uniquely. This paper proposes a design automation technique to provide several circuit design results with different performance to designers so that designers can select an appropriate design among them. The proposed technique is experimentally evaluated with an actual design data, and it is found that the proposed method reduces circuit area in addition to improving RPV. Also the proposed technique is experimentally verified from points of solution quality and computational time.

1. INTRODUCTION

LCD driver circuits are still designed by skilled engineers manually, and CAD environment tailored for LCD panel has not been constructed. LCD panels are often manufactured in different factories even for the same product without design modification due to increase of shipment. Characteristics of transistors are different in each manufacturing factory. When we manufacture LCD panels in a new factory that are compatible or superior in performance to those manufactured in other factories, a new design from scratch is too expensive in design cost. Circuit tuning from the originally designed circuit is a reasonable approach and highly demanded.

In LCD design, display fineness is commonly evaluated as metric, ratio of pixel voltage to video voltage (RPV). The LCD driver circuit charges each pixel to the voltage of a given input video signal. There are several important performance metrics, such as RPV and area, and there is a trade-off among the metrics. Therefore, circuit designers demand a design automation technique that can analyze the trade-off.

This paper proposes a technology migration technique to provide several circuit design results with different performance to designers so that designers can select an appropriate design considering the trade-off relation. Experimental results reveal that technology migration with the proposed technique improves RPV performance and reduces circuit area.

2. LCD COLUMN DRIVER CIRCUIT

2.1. Overview of LCD Driver Circuit

Figure 1 shows an overview of LCD and driver circuit. LCD driver circuits consist of a column driver circuit (Fig. 2) and a row driver circuit, and the column driver circuit is discussed in this paper, since its operation speed is much faster and it affects the overall LCD performance. The column driver circuit provides timing pulses to sampling switches to capture video signal. Figure 2 explains input/output and sub-circuits. Adjacent shift registers output pulses whose timing is different by a half system clock cycle, and these sampling pulses propagate through delay and sampling buffers and finally reach to sampling switches (Fig. 3). Sampling switches get successively on and off according to the given sampling pulses.

The delay buffer adjusts pulse width not to overlap successive

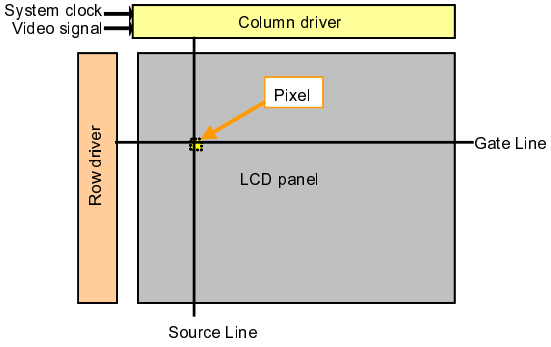


Figure 1: LCD panel and driver circuits.

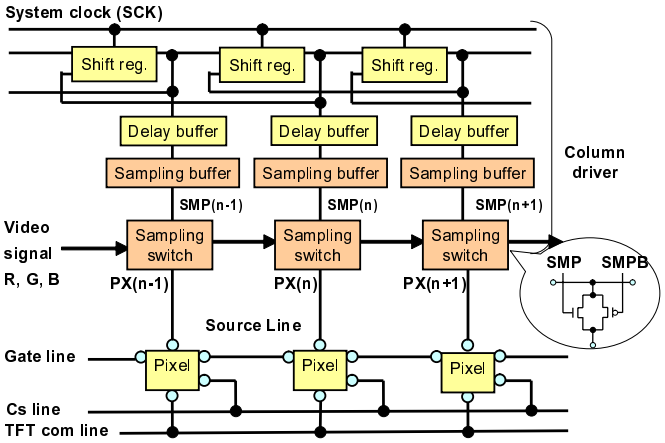


Figure 2: LCD column driver circuit.

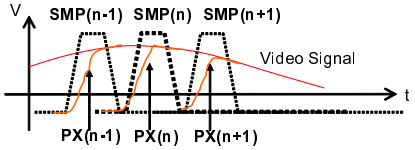


Figure 3: SMP pulses of adjacent drivers and video signal.

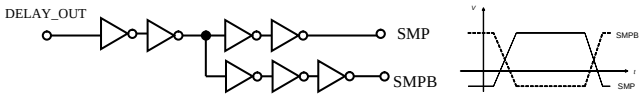
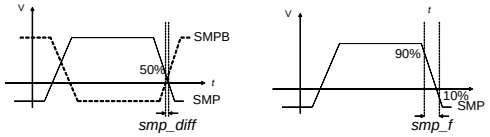


Figure 4: Sampling buffer circuit and sampling pulse shape.



(A) *smp_diff* (B) *smp_f*

Figure 5: Waveform metrics of SMP pulse.

sampling pulses. The sampling buffer (Fig. 4) drives many sampling switches, and hence the load capacitance is large. Therefore sampling buffer should be a kind of cascaded driver[3].

In designing LCD driver circuit, the sampling buffer and the sampling switch are key components to RPV performance, and are focused in this paper. The sampling buffer in Fig. 4 generates SMP and SMPB pulses whose shape should be perfectly opposite for PMOS and NMOS of sampling switches. When SMP is high, the sampling switch is on, and video signals are conducted to pixels.

2.2. Performance metrics and design constraints

A primary design goal is to approach RPV to 100% under process and environmental variability. Smaller area of the designed column driver circuit is desirable.

Figure 5 shows performance metric and design constraint of sampling pulse shape with respect to RPV. Both *smp_diff* and *smp_f* should be minimized so as to capture video signal at the accurate timing.

The above performance metric and constraints must be evaluated under supply voltage variation described in a spec sheet and process fluctuation. In addition, RPV and the load of the sampling buffer depends on the video signal voltage V_{video}, and hence the range of V_{video} should be considered.

2.3. Difficulty in column driver circuit design

We here discuss difficulties in column driver circuit design.

RPV evaluation must be performed under all design constraints, variability conditions and voltage range of V_{dd}, V_{ss}, video, Cs and TFT com line, and hence it requires long CPU time. Thus, it is not practical to perform RPV evaluation for every transistor size change when optimizing sampling switch and driver. We therefore need to divide the design problem into sub-problems to reduce computational cost.

There is a trade-off between pulse waveform metric and circuit area. For example, when enlarging sampling buffer transistors, *smp_f* is reduced and helps to improve RPV, but the circuit area increases. We have to find an appropriate point in the trade-off relation.

When evaluating sampling pulse waveform, we have two choices; circuit simulation and gate delay model, eg. [4]. In sampling buffer design, the important metric *smp_diff* is the delay difference of two paths, and its value is much smaller than the entire delay of the driver circuit. When gate delay estimation is not accurate, the estimation error dominates *smp_diff* value, and

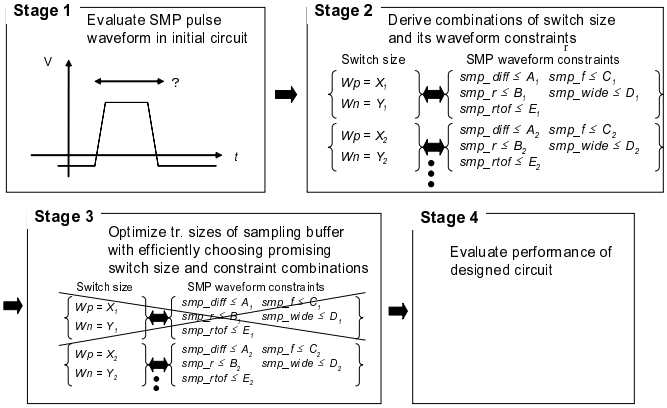


Figure 6: Overview of proposed method.

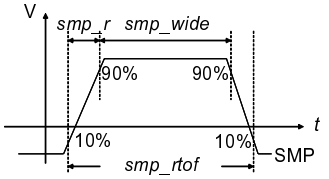


Figure 7: Metrics of sampling pulse shape.

circuit optimization fails. Reference[4][2] reports that gate delay can be modeled in a posynomial expression and then transistor sizing problem can be formulated as a convex optimization problem. However, *smp_diff* includes subtraction and it makes impossible to express *smp_diff* in a posynomial manner. We thus have to solve a non-linear problem that may involve many local optima.

3. PROPOSED METHOD

3.1. Overview

To overcome the difficulties discussed in Sec. 2.3, we propose a design flow that divides the design problem into two sub problems; sampling switch design and sampling buffer design. The size of the design problem is decreased, and hence we can expect CPU time reduction. Figure 6 presents the proposed design flow.

Stage 1 Evaluate sampling pulse waveform of a given initial circuit.

Stage 2 Derive sampling switch size under a given sampling pulse constraint such that RPV is minimized. We list combinations of sampling pulse constraint and sampling switch size[1]. The sampling pulse constraint is set based on the evaluation results in Stage 1.

Stage 3 Design sampling buffer. During the design, non-promising combinations derived in Stage 2 are discarded by solution pruning for computational time reduction.

Stage 4 Evaluate performance of the designed driver circuit.

In the proposed flow, a key is data transfer between Stage 2 and Stage 3, and a definition of metrics for sampling pulse shape is necessary. Three metrics *smp_r*, *smp_wide* and *smp_rtof* shown in Fig. 7 are defined in addition to *smp_diff* and *smp_f* in Fig. 5.

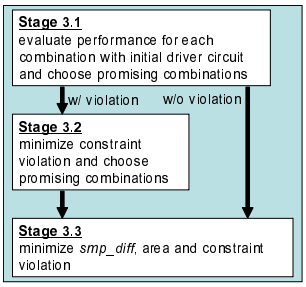


Figure 8: Design process of sampling buffer circuit.

3.2. Sampling Buffer Design

We here explain the proposed method of sampling buffer design in Stage 3.

3.2.1. Optimization Flow

Figure 8 shows optimization flow of sampling buffer consisting of three steps. Stage 3.1 evaluates the performance of the given initial sampling buffer for every combination of sampling switch and pulse shape constraint. We eliminate non-promising combinations. If there is a violation of pulse shape constraint, we quickly optimize the buffer such that the violation is minimized in Stage 3.2. This optimization aims to estimate transistor size increase needed to reduce violation. Promising combinations are selected for Stage 3.3. Stage 3.3 performs transistor sizing to minimize *smp_diif*, *smp_size_sum* and constraint violation.

3.2.2. Objective Function

We explain objective functions obj used in Stage 3.2 and Stage 3.3.

$$obj = excess\_sum \quad (\text{Stage } 3.2) \tag{1}$$

$$obj = obj\_min + excess\_sum \quad (\text{Stage } 3.3) \tag{2}$$

Stage 3.2 minimizes constraint violation and Stage 3.3 optimizes *smp_diif*, *smp_size_sum* and constraint violation with a weighting parameter.

Function *excess_sum* is to compute the sum of constraint violation of sampling pulse waveform and circuit area, and is expressed as follows.

$$\begin{aligned} excess\_sum &= excess(smp\_diff) + excess(smp\_r) \\ &+ excess(smp\_f) + excess(smp\_wide) \\ &+ excess(smp\_rtof) + A \cdot excess(smp\_size\_sum), \end{aligned} \tag{3}$$

where A is a coefficient to balance violations of waveform constraint and circuit area constraint. *excess* is a function to return constraint violation value, and it returns zero when the constraint is satisfied.

Function *obj_min* is expressed as a sum of normalized *smp_diif* and *smp_size_sum* with a weighting coefficient *size_rate*.

$$obj\_min = \frac{smp\_diff}{smp\_diff\_u}(1 - size\_rate) + \frac{smp\_size\_sum}{smp\_size\_sum\_u} \cdot size\_rate, \tag{4}$$

where $0 \leq size\_rate \leq 1$. smp_diff_u and $smp\_size\_sum\_u$ are parameters for normalization. The coefficient *size_rate* enables trade-off analysis between *smp_diif* and *smp_size_sum*, and is set by designers according to the design goal.

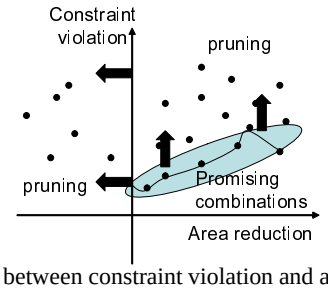


Figure 9: Plots between constraint violation and area reduction for solution pruning in Stage 3.1.

3.2.3. Solution Pruning

To explore efficiently solution space, we execute solution pruning in Stage 3.1 and Stage 3.2. This subsection explains policies of solution pruning. Intensity of solution pruning affects final solution quality and CPU time. When using the proposed technique, we appropriately choose the degree of pruning following the policies below according to the given design time.

1. Smaller switch size desirable
   A sampling buffer drives a number of switches, and hence switch size has a large impact on circuit area and power consumption. Larger switch size also makes buffer design difficult because large load degrades design freedom.
2. Smaller constraint violation preferable
   Larger violation requires more effort, such as area and power, to fix violation.
3. Various switch sizes eligible
   Similar switch sizes result in similar design results, and explored solution space is limited.

Promising combinations of switch size and waveform constraint are selected and poor combinations are discarded, for example, in the following procedure in Stage 3.1. Figure 9 plots constraint violation *excess_sum* versus switch size reduction *switch_size_reduction*, and each dot corresponds to a combination of switch size and pulse constraint. Combinations in the circle are good in the trade-off between violation and switch size, and should be selected.

4. EXPERIMENTAL RESULTS

We implemented the proposed technique with a circuit simulator [6] for performance evaluation, and numerical optimization is performed with Sequential Quadratic Programming[5], because accurate timing evaluation, especially for *smp_diff*, is necessary, and the problem has numerous local optima.

An initial circuit to optimize for technology migration is given. In Stage 1, we first evaluate the initial circuit performance and give the information to sampling switch design in Stage 2. In this experiment, Stage 2 generates 45 combinations with the procedure reported in Ref. [1].

Stage 3.1 evaluates the sampling pulse shapes with the initial sampling buffer. To efficiently save CPU time, we choose promising combinations. Figure 10 plots constraint violation for every combination. According to the selection policy discussed in Section 3.2.3, five combinations (A-E) are selected. The required CPU time is two hours.

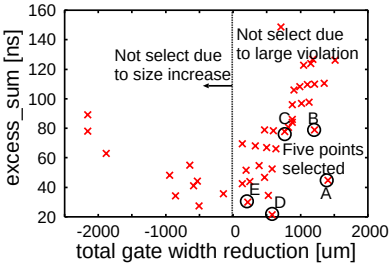


Figure 10: Results of Stage 3.1 and selected combinations.

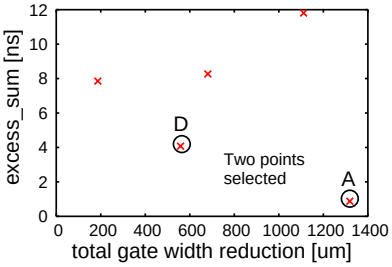


Figure 11: Results of Stage 3.2 and selected combinations.

Table 1: Optimization results in Stage 3.3

metric	A	D
excess_sum	0	2.25
total gate width reduction (μm)	1317.9	473.8

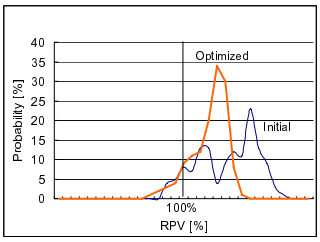


Figure 12: RPV distributions before and after technology migration.

Stage 3.2 minimizes constraint violation for the combinations selected in Stage 3.1. The optimization result is shown in Fig. 11. Two combinations (A, D) are selected; one with small constraint violation and the other is large area reduction. Stage 3.2 needs fourteen hours.

Table 1 lists the optimization results in Stage 3.3. *size_rate* is set to 1. In the case of combination (A), the violation disappears and the total circuit area is reduced by 1318μm. As for combination (D), a slight violation remains, and the area reduction is 474μm. We hence choose combination (A) as the best solution. Stage 3.3 takes ten hours.

Figure 12 shows the distributions of RPV before and after technology migration under process and environmental variation. We can see that the number of samples whose RPV is close to 100% increases, and the performance is improved.

We evaluate the appropriateness of solution pruning. We executed Stage 3.2 and Stage 3.3 for all combinations, and found that combination (A) achieves the largest area reduction in the

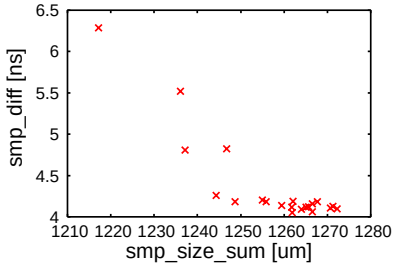


Figure 13: Trade-off between *smp_diff* and *smp_size_sum*.

combinations with zero constraint violation, which means the solution pruning does not degrade the solution quality. We also evaluated the CPU time reduction by solution pruning. Compared with optimization without pruning, the CPU time is reduced from 330 hours to 26 hours by over 90%.

We finally demonstrate the trade-off between *smp_diff* and *smp_size_sum*. We executed optimization in Stage 3.3 varying *size_rate*. Figure 13 shows the result, and we can see that selection of *size_rate* changes the performance as we expected.

5. CONCLUSION

This paper proposes a technology migration technique for LCD driver circuits. To efficiently perform transistor sizing, we divide the optimization problem into two sub-problems. Sampling switch design first derives combinations of sampling switch and sampling pulse constraint. In the following, sampling buffer is optimized for the given combinations. To reduce computational time without degrading solution quality, solution pruning based on several policies is executed in sampling buffer design process. Experimental results show that the proposed technique successfully execute technology migration improving performance in RPV and area. We also confirm that the problem division and solution pruning enable sampling buffer design in a practical time.

6. REFERENCES

[1] S. Takahashi, S. Tsukiyama, M. Hashimoto and I. Shirakawa, "A Sampling Switch Design for Liquid Crystal Displays," in *Proc. IEEE Int'l Region 10 Conf.*, 1C-03.3, 2005.

[2] S. Boyed and L. Vandenberghe, "Convex Optimization" Cambridge University Press 2004, 2004.

[3] H. B. Bakoglu, "Circuits, Interconnections, and Packaging for VLSI," Addison-Wesley Publishing Company, 1990.

[4] K. Kasamsetty, M. Ketkar, and S. S. Sapatnekar, "A new class of convex functions for delay modeling and their application to the transistor sizing problem," *IEEE Trans. on CAD*, Vol. 19, No. 7, pp.779-788, July 2000.

[5] C. Laurence, J. L. Zhou and A. L. Tits, "User's Guide for CFSQP Version 2.5: A C Code for Solving (Large Scale) Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying All Inequality Constraints," April 1997.

[6] Silvaco Inc., "SmartSpice User's Manual,", 2005.