**Slide 1: Title**

Thank you for the introduction. I am presenting this paper on behalf of my colleague Mr. Mulong Luo. This is a joint research work of UC San Diego and Samsung Electronics.

**Slide 2: Outline**

My talk is structured as follows. First, I provide background and motivation of the work. Second, I explain our crosstalk-aware layout optimization methods, followed by the testcase generation and experimental setup and results. Finally, I will conclude my talk.

**Slide 3: Introduction**

DRAM interconnect channels are regions used to connect the IO pad and DRAM arrays. These channels are narrow and long.

The channels contain many parallel wires, which can introduce large crosstalk. So, we must assign tracks to these wires so that the crosstalk does not cause large delay uncertainty. The manual, non-automated design is still the dominant methodology for the design of the DRAM channels. Manual track assignment might be far from optimal.

We propose an automated DRAM channel layout optimizer to minimize the crosstalk effects.

**Slide 4: Related works (Word count: 87)**

We review previous works in the areas of crosstalk-aware analysis and design.

In crosstalk-aware analysis, Xiao00 proposes analytical modeling of crosstalk-induced delay and noise. Gross98 and Sato00 describe how the alignment of aggressor impacts crosstalk noise and delay on the victim.

In crosstalk-aware design, Yu09 propose a swizzling pattern to reduce crosstalk-induced delay.

Gao93 propose crosstalk-aware MILP-based detailed routing.

However, no existing works integrate simultaneous crosstalk-aware analysis and automated design.

**Slide 5: Our Contribution (Word count： 80)**

Our key contributions are:

First, we develop an accurate closed-form analytical delay uncertainty calculator.

Second, we integrate swizzling of signals in the channel to exploit signal correlation to reduce worst-case delay uncertainties. We propose MILP-based segment optimization and pair-swapping segment optimization methods.

We achieve up to 29% reduction of maximum weighted delay uncertainty compared to conventional methods.

**Slide 6: Outline (word count: 10)**

Now, I explain our crosstalk-aware layout optimizer.

**Slide 7: Problem Statement (Word count: 116)**

The inputs to our optimizer are: set of tracks T, set of segments G, set of signals S, and criticality classes of signals.

Criticality denotes the priority of signal. For example, clock signal has the highest-criticality. Other inputs include design rules and inter-buffer lengths for each class.

Our objective is to minimize the max weighted delay uncertainty among all signals in different classes.

**Slide 8: Problem complexity (Word count: 107)**

This problem is very complex. The size of the solution-space of exhaustive search is an astronomical number T permute S to the power of G. For example, a small testcase with 4 signals, 4 tracks on 4 segments have 331K possibilities.

To reduce the solution-space, we perform segment-by-segment optimization. Thus, we reduce the size to T permute S times G. For the same example, we now have 96 possibilities.

However, we pessimistically assume that the segments are not coupled. This is achieved by assigning infinite distance between wires.

**Slide 9: Segment-by-segment optimization (Word count: 100)**

The figure explains why the assumption is pessimistic. When distance increases, the coupling cap will decrease, thus, the voltage change due to crosstalk on this segment will increase, which leads to increased delay uncertainty.

This assumption can lead to suboptimality, but according to our experiments, the suboptimality is negligible. We use the same example from the previous slide; additional parameters are listed here. The optimal results give maximum delay uncertainty of 7.16ps, whereas with segment-by-segment it is 7.19ps. Without any segment-by-segment optimization, we have 9.57ps. This shows the suboptimality of only 0.4 percent.

**Slide 10: Overview of Crosstalk-Aware Layout Optimization (Word count: 71)**

Given testcase specifications, we perform segment-by-segment optimization by either a MILP-based segment optimization or a pair swapping segment optimization.

The optimization is guided by out accurate and fast delay uncertainty calculator.

**Slide 11: MILP-based Segment Optimization: Notations (Word count: 87)**

I describe notations used in our MILP.

D\_i is the delay uncertainty of signal at the input of current segment.

Delta\_di\_i’ is the estimated delta delay uncertainty of signal si at the end of the current segment when signal si’ is its neighbor.

lambda\_l is a testcase-specific variable that represents the weight of class.

The output variables are q\_ik and p\_ii’, which are binary indicators for the assignment of signals and neighbors.

**Slide 12: Basic constraints (Word count: 91)**

Now I explain our constraints. This constraint ensures that each signal occupies exactly one track.

Similarly, the second constraint forces that each track can only be occupied by at most one signal.

If signal I is on track k and signal I’ is on track k-1 or vice versa, then this constraint enforces that they are neighbors.

The last constraint makes sure that for |S| tracks, there will be at most |S|-1 pairs of neighbors.

**Slide 13: MILP-based segment optimization (Word count: 83)**

We first pre-calculate the estimated delta delay uncertainty of all combinations of signal pairs using our calculator.

Then, we assign the pre-calculated values in our formulation as shown in this constraint.

The objective is to minimize the max weighted delay uncertainty.

As an example, given 4 signals, our optimizer assigns the signals on the current segment. When it assigns the signals, it considers the estimated delta delay uncertainties of all pairs to minimize the delay uncertainty.

**Slide 14: Decomposition for Scalability (Word count: 154)**

Our MILP can handle only up to 30 signal- and 30 track-sized channels. For the testcases with hundreds of signals, we propose the following decomposition method.

Let’s say the subset size we decompose is V. For the first segment, we ensure that the size is less than or equal to V.

For the subsequent segments, we offset the boundaries by V/2 so that signals from different subsets in the previous segment can be mixed to exploit the signal correlations.

Now, we solve several MILP instances in parallel for each segment, which reduces runtime significantly.

**Slide 15: Overview of Crosstalk-aware layout optimization (Word count: 12)**

Now, I explain our pair-swapping segment optimization.

**Slide 16: Pair-swapping Segment Optimization (Word count: 115)**

For a large testcase, this method is faster and gives better max weighted delay uncertainty than MILP-based optimization with decomposition.

The main idea of pair swapping is to swap signals with maximum weighted delay uncertainty with other signals.

First, we sort all the signals by the increasing order of delay uncertainty.

Second, we swap the signals with max-weighted delay uncertainty and the signals with min-weighted delay uncertainty. Then, check the max-weighted delay uncertainty. If no improvement is seen, we revert the swapping. Otherwise we keep the swap.

We repeat the steps 1, 2 until there is no improvement.

**Slide 17: Overview of crosstalk-aware layout optimization (Word count: 12)**

Now, I describe our delay uncertainty calculator.

============================================

**Slide 18: Delay Uncertainty Calculator (words: 107)**

Long runtime is a well-known limitation of SPICE simulations. To overcome this limitation, we have developed a fast closed-form delay uncertainty calculator.

For a given aggressor-victim pair, we generate noise waveform induced by crosstalk using a 2pi circuit. We then measure the impact of noise on dynamic delay change of the victim signal. This is called delay change curve proposed in Sato03. Finally, we get the delay uncertainty of victim signal from DCC. For multiple signals, we calculate delay uncertainties for all neighboring pairs and superpose them.

**Slide 19: Accuracy of Delay Uncertainty Model (words: 99)**

Here, we compare our model and the model in Gupta04.

We generate 300 random swizzling patterns for a simple testcase with 5 signals, 5 tracks and a 8000um channel divided into 8 segments. We then correlate ranks with SPICE simulation results. The left figure shows rank correlation between our model and SPICE. And, right figure shows the rank correlation between Gupta04 model and SPICE. We see that our model shows much better correlations to the SPICE results.

**Slide 20: Outline**

Now, I explain how we generate artificial, but realistic testcases.

**Slide 21: Testcase Generation: General Inputs (words: 145)**

To our knowledge, there is no public benchmark to study DRAM channel routing problems. So, we develop a testcase generator guided by an engineer in a leading DRAM manufacturer. There are three types of inputs to the generator; general inputs, class-specific inputs and signal-specific inputs.

Here, I will discuss the general inputs. Channel length, channel width, number of signals, number of tracks and number of segments are required as shown this figure. We also need a probability table that correlates signals between all pairs of classes. For example, a value of 1 means in Row 2 Column 3 that all signals in class 0 are correlated with all signals in class 1. We assume if any two signals are correlated, their timing window could be overlapped. Supply voltage and clock period are also provided as general inputs to our testcase generator.

**Slide 22: Testcase Generation: More Inputs (words: 136)**

Here is the list of class-specific inputs which include the number of signals in each class, ground cap and resistance per micron. There are different coupling capacitances per micron among signals in different classes since the signals in different classes follow different width and spacing rules. In our optimization, the distance between two consecutive buffers is fixed. We are also given the input pin capacitance and the drive resistance of buffers as inputs.

Last, there are signal-specific inputs. It includes load cap., resistance at signal input, input slew and correlation with other signals. Correlation between any two signals is determined by the probability table that I described in previous slide. If any two signals are correlated, we assume their timing windows are overlapped and the signals switch in different directions.

**Slide 23: Outline**

Now, I describe our experimental setup and results.

**Slide 24: Experimental Setup (words: 154)**

We use a fixed channel length 8000um for all testcases. We use signals with five classes. Pitch, width, space and buffer locations are shown in this table. Class 0 is the highest criticality class. Other signal-specific inputs are shown here.

We conduct a total of four experiments. In Experiments 1, 2 and 3, we study the impact of different input conditions on delay uncertainty using small testcases. We vary the number of signals, the number of tracks and the correlation between signals, and observe how our optimizer delivers different layout solutions.

In Experiment 4, we compare MILP, pair-swapping and signal permutation methods with larger and more realistic testcases. I will explain the details as part of the experimental results.

**Slide 25: Experiment 1: Impact of Number of Signals and Tracks (words: 163)**

We vary the number of signals and tracks to see the impact of these on delay uncertainty.  
In this experiment, we use the weights of the five classes as 10, 6.7, 4, 2, 1. 10 is used for the highest-criticality class and 1 is used for the lowest-criticality class. The number of signals for each class is the same. For example, testcase E1T1 has two signals in each class.

These are the results of testcase E1T1. The lower-left figure shows the change of delay uncertainty in the Y-axis and channel distance in the X-axis for each class. We see the signals in the higher-criticality class achieve smaller delay uncertainty.

This figure shows the final layout solution. The thickness represents delay uncertainty. I want to note that the signals in the highest-criticality class are mostly on the boundary of the channel to minimize the delay uncertainty.

**Slide 26: Experiment2: Impact of Percentage of Signals in Each Class (words: 96)**

Here, we use a fixed number of signals and tracks which is 20 and only vary the percentage of signals in different classes. We use the same weights used in Experiment1. The table shows the priority and number of signals together with the maximum delay uncertainty of each class and objective function value.

We see that the objective value reduces if the percentage of lower-criticality signals increases. Also, when we change class B to lower priority, the max delay uncertainty of signals in class A reduces accordingly.

**Slide 27: Experiment3: Impact of Correlation of Signals (words: 115)**

In this experiment, we set the number of signals and tracks as five. We generate two testcases E3T1 and E3T2. Both have four signals in the higher-criticality class and one signal in the lower-criticality class. In E3T1, signals in different classes are fully correlated, but in E3T2, signals in different classes are not correlated.

Below figures show the layout solutions. Orange-colored signal is in the lower-criticality class. We observe that the orange-colored signal in E3T2 is routed in the middle of channel. This maximizes the mutual shielding effect to minimize delay uncertainty. As a result, we see that the maximum delay uncertainty of signals in E3T1 is larger than that in E3T2.

**Slide 28: Experiment4: MILP vs. Pair-Swapping vs. Signal Permutation (1) (words: 109)**

To evaluate our methods with more realistic testcases, we generate larger four testcase T2 to T5. The number of tracks and the number of signals in each class are shown here. Testcases T3 and T5 have some empty tracks to study the impact of the empty tracks. The size of decomposition subset of MILP-based method is 20.

Overall, pair-swapping method shows better results than signal permutation. It shows up to 29% reduction of maximum weighted delay uncertainty. With empty tracks in T3 and T5, delay uncertainty is reduced by up to 19.9%. Also, pair-swapping run faster than MILP-based method.

**Slide 30: Outline**

Now, I conclude my talk.

**Slide 31: Conclusion**

In this work, we propose a DRAM routing channel optimizer to specifically target the layout design of long, resource-constrained channels in modern DRAM products.

Our optimizer is signal criticality-aware, and minimizes a maximum weighted delay uncertainty.

We achieve up to 29% reductionof maximum weighted delay uncertainty compared to a traditional track permutation methodology.

Our ongoing work includes flexible buffer location and use of inverters to further reduce delay uncertainty.