(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 January 2023 (26.01.2023)

WIPO | PCT

(10) International Publication Number
WO 2023/003918 A2

(51) International Patent Classification:
G06F 30/18 (2020.01) *G06N 20/00* (2019.01)

(21) International Application Number:
PCT/US2022/037648

(22) International Filing Date:
19 July 2022 (19.07.2022)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/203,455 23 July 2021 (23.07.2021) US

(71) Applicant: **D2S, INC.** [US/US]; 4040 Moorpark Ave, Suite 250, San Jose, CA 95117 (US).

(72) Inventors: **FUJIMURA, Akira**; 4040 Moorpark Ave, Suite 250, San Jose, CA 95117 (US). **SHIRALI, Nagesh**; 1843 Fumia Place, San Jose, CA 95131 (US). **ORIORDAN, Donald**; 4040 Moorpark Ave, Suite 250, San Jose, CA 95117 (US).

(74) Agent: **ADELI, Mani**; Adeli LLP, P.O. Box 516, Pacific Palisades, CA 90272 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,

(54) Title: METHODS AND SYSTEMS TO DETERMINE PARASITICS FOR SEMICONDUCTOR OR FLAT PANEL DISPLAY FABRICATION

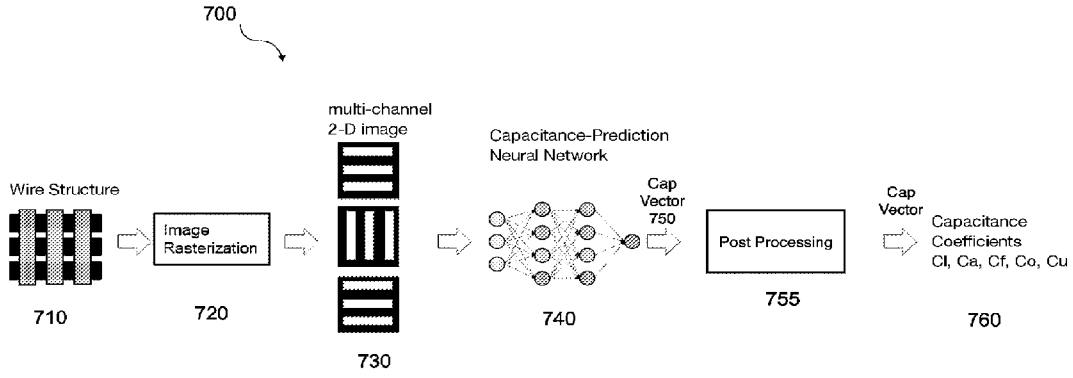

FIG. 7

(57) Abstract: Some embodiments provide a method for calculating parasitic parameters for a pattern to be manufactured on an integrated circuit (IC) substrate. The method receives a definition of a wire structure as input. The method rasterizes the wire structure (e.g., produces pixel-based definition of the wire structure) to produce several images. Before rasterizing the wire structure, the method in some embodiments decomposes the wire structure into several components (e.g., several wires, wire segments or wire structure portions), which it then individually rasterizes. The method then uses the images as inputs to a neural network, which then calculates parasitic parameters associated with the wire structure. In some embodiments, the parasitic parameters include unwanted parasitic capacitance effects exerted on the wire structure. Conjunctively, or alternatively, these parameters include unwanted parasitic resistance and/or inductance effects on the wire structure.

WO 2023/003918 A2

[Continued on next page]

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

METHODS AND SYSTEMS TO DETERMINE PARASITICS FOR SEMICONDUCTOR OR FLAT PANEL DISPLAY FABRICATION

BACKGROUND

[0001] Three common types of charged particle beam lithography are unshaped (Gaussian) beam lithography, shaped charged particle beam lithography, and multi-beam lithography. In all types of charged particle beam lithography, charged particle beams shoot energy to a resist-coated surface to expose the resist.

[0002] In the production or manufacturing of semiconductor devices, such as integrated circuits, optical lithography may be used to fabricate the semiconductor devices. Optical lithography is a printing process in which a lithographic mask or photomask manufactured from a reticle is used to form patterns on a substrate such as a semiconductor or silicon wafer to create the integrated circuit. Other substrates could include flat panel displays or even other reticles. Also, extreme ultraviolet (EUV) or X-ray lithography are considered types of optical lithography. The reticle or multiple reticles may contain a circuit pattern corresponding to an individual layer of the integrated circuit, and this pattern can be imaged onto a certain area on the substrate that has been coated with a layer of radiation-sensitive material known as photoresist or resist. Once the patterned layer is created the layer may undergo various other processes such as etching, ion-implantation (doping), metallization, oxidation, and polishing. These processes are employed to finish an individual layer in the substrate. If several layers are required, then the whole process or variations thereof will be repeated for each new layer. Eventually, a combination of multiples of devices or integrated circuits will be present on the substrate. These integrated circuits may then be separated from one another by dicing or sawing and then may be mounted into individual packages. In the more general case, the patterns on the substrate may be used to define artifacts such as display pixels or magnetic recording heads.

[0003] In the production or manufacturing of semiconductor devices, such as integrated circuits, maskless direct write may also be used to fabricate the semiconductor devices. Maskless direct write is a printing process in which charged particle beam lithography is used to form patterns on a substrate such as a semiconductor or silicon wafer to create the integrated circuit. Other substrates could include flat panel displays, imprint masks for nano-imprinting, or even reticles. Desired patterns of a layer are written directly on the surface, which in this case is also the substrate. Once the patterned layer is created the layer may undergo various other processes such as etching, ion-implantation (doping), metallization, oxidation, and polishing. These processes are employed to finish an individual layer in the substrate. If several

layers are required, then the whole process or variations thereof will be repeated for each new layer. Some of the layers may be written using optical lithography while others may be written using maskless direct write to fabricate the same substrate. Eventually, a combination of multiples of devices or integrated circuits will be present on the substrate. These integrated circuits are then separated from one another by dicing or sawing and then mounted into individual packages. In the more general case, the patterns on the surface may be used to define artifacts such as display pixels or magnetic recording heads.

[0004] Modeling parasitic effects in an IC design is very important. Parasitic effects refer to unwanted parasitic capacitance, resistance and inductance on the components (e.g., on wire segments) in the IC design. The different parasitic effects can impact circuit delay, energy consumption and power distribution. They can also introduce noise sources and other effects that impact reliability. To evaluate the effect of interconnect parasitics on circuit performance, they need to be accurately modeled.

[0005] Different techniques have evolved over time to model parasitics, including unwanted capacitance, resistance and inductance, as manufacturing techniques have become more complex. However, in recent years, the modeling and extraction of parasitics has become more difficult at smaller process geometries and newer process nodes. Many of the difficulties stem from the increasing effects of manufacturing process variations and other types of manufacturability issues at smaller geometries. The existing techniques are also relatively slow in computing the parasitic parameters.

BRIEF SUMMARY

[0006] Some embodiments provide a method for calculating parasitic parameters for a pattern to be manufactured on an integrated circuit (IC) substrate. The method receives a definition of a wire structure as input. The method rasterizes the wire structure (e.g., produces pixel-based definition of the wire structure) to produce several images. Before rasterizing the wire structure, the method in some embodiments decomposes the wire structure into several components (e.g., several wires, wire segments or wire structure portions), which it then individually rasterizes. The method then uses the images as inputs to a neural network, which then calculates parasitic parameters associated with the wire structure. In some embodiments, the parasitic parameters include unwanted parasitic capacitance effects exerted on the wire structure. Conjunctively, or alternatively, these parameters include unwanted parasitic resistance and/or inductance effects on the wire structure.

[0007] Some embodiments provide a method for training a neural network to extract parasitic capacitance from a semiconductor design. This method receives as input the semiconductor design that includes several wire structures. The method performs a rasterization operation that rasterizes each wire structure into several 2-D images. For each wire structure, the method uses these images as input to a machine trained network (e.g., a neural network) that generates one or more curvilinear shapes to represent the wire structure. The method then uses the set of the curvilinear shapes for each wire segments to train the neural network.

[0008] The preceding Summary is intended to serve as a brief introduction to some embodiments of the invention. It is not meant to be an introduction or overview of all inventive subject matter disclosed in this document. The Detailed Description that follows and the Drawings that are referred to in the Detailed Description will further describe the embodiments described in the Summary as well as other embodiments. Accordingly, to understand all the embodiments described by this document, a full review of the Summary, Detailed Description, the Drawings and the Claims is needed. Moreover, the claimed subject matters are not to be limited by the illustrative details in the Summary, Detailed Description, and Drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The novel features of the invention are set forth in the appended claims. However, for purposes of explanation, several embodiments of the invention are set forth in the following figures.

[0010] **FIG. 1** illustrates a digital design flow, as known in the art.

[0011] **FIG. 2** illustrates an example bus structure, as known in the art.

[0012] **FIG. 3** illustrates a detailed flow to calculate parasitics, as known in the art.

[0013] **FIGs. 4A and 4B** illustrate a wire structure, as known in the art.

[0014] **FIG. 5** illustrates a flow to calculate parasitics, as known in the art.

[0015] **FIGs. 6A and 6B** illustrate flows to calculate parasitics, in accordance with some embodiments.

[0016] **FIG. 7** illustrates a flow to calculate parasitics, in accordance with some embodiments.

[0017] **FIG. 8** illustrates a neural network architecture, in accordance with some embodiments.

[0018] **FIG. 9** illustrates training data generation, in according with some embodiments.

[0019] **FIG. 10** illustrates a flow to calculate parasitics using tiles, in accordance with some embodiments.

[0020] **FIG. 11** illustrates tiled data, in accordance with some embodiments.

[0021] **FIG. 12** illustrates a flow in training a neural network to generate tiles, in accordance with some embodiments.

[0022] **FIG. 13** illustrates a neural network to calculate parasitics using tiles, in accordance with some embodiments.

[0023] **FIGs. 14 and 15** are schematics of GPU system diagrams, in accordance with some embodiments.

[0024] **FIG. 16** illustrates a capacitance matrix for a 3x3 bus structure, in accordance with some embodiments.

DETAILED DESCRIPTION

[0026] In the following detailed description of the invention, numerous details, examples, and embodiments of the invention are set forth and described. However, it will be clear and apparent to one skilled in the art that the invention is not limited to the embodiments set forth and that the invention may be practiced without some of the specific details and examples discussed.

[0027] Semiconductor wiring, more commonly called interconnect, forms a complex 3-D geometry that introduces unwanted parasitic capacitance, resistance and inductance. Dealing effectively with these unwanted parasitic effects is a process that typically takes circuit and mask designers multiple iterations in order to create manufacturable designs that meet specifications, yield well, and offer good reliability. Hence, in electronic design automation (EDA), there is a need to properly extract and model parasitic effects (e.g., unwanted capacitance, inductance and/or resistance effects) in the IC design accurately. Different techniques have evolved over time to model parasitics, including unwanted capacitance, resistance and inductance, as manufacturing techniques have become more complex.

[0028] This extraction/modeling step is becoming more and more difficult at smaller process geometries/newer process nodes. Many of the difficulties stem from the increasing effects of manufacturing process variations and other types of manufacturability issues at smaller geometries. Over the years, even though advances in processing technology have reduced the effect of resistance, and low-k dielectric materials have reduced the effect of capacitance, the parasitic effects have continued to remain dominant or have increased in dominance due to the continued scaling down of feature sizes (wire widths, etc.).

[0029] The different parasitic effects can impact circuit delay, energy consumption and power distribution. They can also introduce noise sources and other effects that impact reliability. To evaluate the effect of interconnect parasitic effects on circuit performance, they need to be accurately modeled. **FIG. 1** shows a simplified digital design flow used conventionally, in which parasitics are extracted from a circuit layout in a back end portion of the flow, and accounted for in gate level simulations in a front end portion of the flow. Typically, interconnect parasitics will affect delay/timing, leading to changes in the gate level netlist, requiring another iteration through floorplanning and/or place and route, and resulting with modified circuit layout. More complex digital design flows may replace the back end portion with other steps, such as virtual prototyping, power-grid synthesis, placement, power routing, clock tree synthesis (CTS), post-CTS optimization, routing, post-route timing and

signal-integrity optimization, and finally signoff extraction, timing signal-integrity, and power sign off. Parasitics must also be accounted for in these additional design steps.

[0030] Analog design flows also require detailed simulations after the layout has been completed and parasitics have been extracted, with the parasitics affecting the simulation results in a way that requires a change of layout. More complex analog design flows also involve a form of layout prototyping, floorplanning, placement and routing, and attempt to account for layout-dependent effects (LDE) and density-gradient effects (DGE). In both analog and digital flows, routing needs to be aware of multiple-patterning, where pattern density is addressed by separating a layout into lower densities to be exposed separately, due to its impacts on printability. As in the digital flows, the various steps in analog design flows require that parasitics are accurately accounted for.

[0031] In both analog and digital flows, multiple iterations are typically required until a layout is achieved that meets timing, power, performance and area design constraints, in the presence of parasitics. Detailed simulations need to be repeated not only with nominal process condition parasitics, but also with parasitic variations representing the various manufacturing process corners, in order to ensure these circuit-level metrics are met across manufacturing process variations.

[0032] While the following description focuses primarily on capacitance extraction techniques, the methods described herein also apply to extraction of resistance and inductance.

[0033] FastCAP is an existing three-dimensional capacitance extraction program that computes self and mutual capacitances between ideal conductors of arbitrary shapes, orientations, and sizes. **FIG. 2** provides an example to illustrate the operation of such an extraction program. It illustrates a bus structure 200 for which its parasitic capacitance has to be extracted. As shown, the bus structure 200 includes four conductors each with six faces that are represented as patches. Conductors are broken into sections based on where they overlap. Input files for FastCap specify the discretization of conductor surfaces into panels, where edges are more finely meshed for accuracy. For the example in **FIG. 2**, FastCAP produces a 4x4 capacitance matrix, shown in the table below.

CAPACITANCE MATRIX, picofarads				
	1	2	3	4
1%GROUP1 1	247.8	-85.01	-48.53	-48.53
2%GROUP1 2	-85.01	247.8	-48.53	-48.54
3%GROUP1 3	-48.53	-48.53	247.9	-84.99
4%GROUP1 4	-48.53	-48.54	-84.99	247.9

After solving the Maxwell's equations for the given structure, a symmetric capacitance matrix is then produced as output by the field solver, listing the conductor self-capacitances along the main diagonal, with the off-diagonal terms being the coupling capacitances among the various conductors.

[0034] Various methods may be used in technology pre-characterization in which several structures are simulated using a field solver, from which coupling capacitance coefficients are eventually computed. **FIG. 3** illustrates that at a high-level a typical pre-characterization process starts by constructing several multi-layer 2-D circuit wire structures 300, containing conductor wires of various widths and spacings. These 2-D wire structures are then coupled with wire height information from a process technology file, and extruded (by an extrusion process 302) to form 3-D structures.

[0035] The 3-D structures are then converted into a form for the field solver 304 to process. For instance, the 3-D structures are converted into N conductors with a set of panels, which are then consumed by the field solver 304, which produces a NxN capacitance matrix 306. The capacitance matrix is filtered by a filter process 308 to produce a set of self-capacitance values and a set of coupling-capacitance values. These capacitance values are then post processed into component values, i.e., capacitance coefficients.

[0036] In a paper entitled "Analysis and Justification of a Simple, Practical 2 1/2D Capacitance Extraction Methodology" by Cong et al., five foundations for a capacitance extraction methodology are presented.

- The first foundation is that "ground, and neighboring wires on the same layer have significant shielding effects. Thus, both must be considered for accurate modeling."
- The second foundation is that "coupling between wires in layer I + 1 and wires on layer i - 1 is negligible when the metal density on layer i exceeds a certain threshold."
- The third is that "during capacitance extraction for wires on layer i, layers i +/- 2 can be treated as ground planes with negligible error. There is no need to look beyond layers i +/- 2."
- The fourth is that "coupling analysis to wires in the same layer need only consider nearest neighbors independently, with the widths of same-layer neighbor wires having negligible effects on the coupling."

- The fifth and final foundation is that "the joint interaction of layers i-1 and i+1 on layer i is negligible, therefore corrections for orthogonal crossovers and crossunders can be performed incrementally."

[0037] These foundations as well as nomenclature from the above-mentioned paper are used in the parasitic extraction methodology of some embodiments of the invention. For example, **FIG. 4A** shows a single-layer structure allowing lateral (Cl), area (Ca) and fringe (Cf) capacitance coefficients to be extracted for a wire of width W, with same-layer neighbor spacing S, on layer i. The structure on the left shows three wires of identical width W, while the structure on the right shows three slightly different wires of identical width W' (W-prime), i.e., wires that are very slightly different in width from those of the structure on the left.

[0038] During pre-characterization, 2-D bus structures corresponding to both patterns are created. Wire heights from the process technology file are used to produce 3-D structures from the 2-D structures. The 3-D structures are then meshed to create a series of 2-D surface panels, and the panel information is used as input to the field solver. The 3-D structures are simulated by a field solver (e.g., FastCAP), resulting in two capacitance matrices. Simultaneous equations are constructed relating the capacitance matrix values to Cl, Ca and Cf capacitance component values, and solved to produce Cl, Ca and Cf component values for that particular wire width and spacing, i.e., W, S pair. The approach is then repeated for various values of W, S.

[0039] **FIG. 4B** shows a plan-view of the geometric structure for computing crossover capacitances. The left side of this figure contains a 3x3 bus crossing structure 420, and its right side includes a 3x2 bus crossing structure 425. Both structures are extruded to 3-D using the wire heights from a process technology file, and then converted to surface panels. The surface panel information is then used as input and solved (independently) by a field solver (FastCAP), which produces 3x3 and 3x2 capacitance matrices respectively. The resulting capacitance matrices are then post-processed to produce values for crossover capacitance.

[0040] Under this approach, the process is repeated with different values of crossover wire width and spacing Wc and Sc, along with different values of layer-of-interest width W and spacing S. The various capacitance matrices are then post-processed to allow for the crossover capacitance coefficient to be determined as a function of the 4-tuple (W, S, Wc, Sc). A similar approach is used to determine crossunder capacitance (using layer i-1 in lieu of layer i+1). Other approaches may use 3-layer bus crossing structure, or other structures, along with appropriate post-processing techniques to determine capacitance coefficients.

[0041] Using a variety of different values for W, S, Wc, Sc, a variety of values are then computed for Cl, Ca, Cf, crossover capacitance coefficient Co, and crossunder capacitance coefficient Cu. These values are then used to compute lookup tables, which allow for Cl, Ca, Cf, Co, Cu to be looked up during a later capacitance extraction phase, as a value of W, S, Wc, Sc. The lookup tables are stored as part of the pattern library for the extractor.

[0042] During the capacitance extraction phase, the geometric parameters for wire segments of an IC design of interest are determined, and the lookup tables in the pattern library are examined to find the capacitance component coefficient values. Linear interpolation in W, and 1/S is used when the values of the wires encountered during the capacitance extraction phase do not exactly match those used in the lookup table generation during the pre-characterization phase.

[0043] Note that this and other approaches have been related to geometric parameters. Models and tables are stored with the pattern library during pre-characterization as a function of geometric parameters, such as widths and spacings. During the extraction phase, a layout is decomposed to a set of geometric parameters (more widths and spacings), and the models/tables from the pre-characterization phase are consulted/looked up, with linear interpolation as necessary, to compute capacitance values.

[0044] To model the effects of parasitic capacitance over process generations, methods to calculate parasitic capacitance from layout data have evolved from 1-D, 2-D, 2.5-D and all the way to fully 3-D based solutions in order to meet the requisite accuracy.

[0045] Regardless of accuracy level, capacitance extraction is generally performed in two phases. **FIG. 5** illustrates a conventional flow for performing capacitance extraction. A first phase known as "pre-characterization" requires process technology information but does not require the layout for the actual IC to be extracted. This first phase is performed once per process technology node. This first phase is depicted above the dashed line 500 in **FIG 5**. The second phase, called an extraction phase, requires an actual IC layout design database for the chip for which parasitic parameters are to be extracted, along with information produced during the pre-characterization phase. This phase is shown below the line 500 in **FIG. 5**. This phase is required once per IC design and produces a parasitics file or database as output.

[0046] During pre-characterization, CPU intensive but highly accurate field solvers are used to determine the capacitances for particular structures. The resulting capacitances are then post-processed in conjunction with particular capacitance models, resulting in various sets of model parameters or lookup tables representative of the manufacturing process technology. The models and/or lookup tables are then stored as the output of the pre-characterization phase.

The parameterized models and/or lookup tables stored during the pre-characterization phase are then combined with geometry information regarding the IC design during the capacitance extraction phase.

[0047] A typical Reticle Enhancement Technology (RET) method has Optical Proximity Correction (OPC) verification to identify and correct hot spots. A hot spot is an area requiring ideal conditions to print properly and therefore is not resilient to manufacturing variation, or in some cases would not print properly even in ideal conditions. Hot spots lead to poor yield. Inverse Lithography Technology (ILT) is one type of OPC technique. ILT is a process in which a pattern to be formed on a reticle is directly computed from a pattern that is desired to be formed on a substrate such as a silicon wafer. This may include simulating the optical lithography process in the reverse direction, using the desired pattern on the substrate as input. ILT-computed reticle patterns may be purely curvilinear—i.e., completely non-rectilinear—and may include circular, nearly circular, annular, nearly annular, oval and/or nearly oval patterns. Numerous studies and wafer results have shown that ILT – in particular, unconstrained curvilinear ILT – can produce the best results in terms of wafer-pattern fidelity and process window.

[0048] In critical or very dense IC designs, it is essential to model the parasitic capacitance values as accurately as possible so that any effects on timing (performance) and power consumption are taken into accounted fully. Some embodiments achieve such capacitance extraction accuracy by incorporating manufacturing process simulators capable of producing curvilinear shapes, with proximity effects included, directly into both the pre-characterization and extraction phases of capacitance extraction.

[0049] Curvilinear shapes more closely resemble the manufactured shapes of components (e.g., wires) in the IC design. Hence, using curvilinear shapes to perform parasitic extraction improves the accuracy of the extracted parasitic values. **FIG. 6A** illustrates an example of a manufacturing process simulator that produces and uses curvilinear shapes during pre-characterization. The manufacturing process simulator in this example includes an RET 605 that produces curvilinear 2D shapes 610 from 2D shapes 600 using information from an IC layout database that informs how it should produce the curvilinear shapes 610. The manufacturing process simulator in this example also includes a semiconductor process model 615, which is a set of parameters that describe the semiconductor manufacturing process. This semiconductor process model 615 contains process models such as the type of light source used for lithography, the wavelength of the light used, etc.

[0050] The produced 2-D curvilinear shapes 610 are then fed to the 3-D extrusion and meshing process 616, which then uses these shapes and information from the process technology file 618 to produce precise meshed 3-D shapes 620. These 3-D shapes 620 are then provided as input to a field solver 622. The description of the 3-D shapes 620 is more accurate than what would be produced using traditional methods. This, in turn, results in significantly more accurate capacitance values 624 being produced by the field solver 622.

[0051] The manufacturing process simulators used in some embodiments simulate a variety of the detailed defects of manufacturing, allowing for detailed plan views of the shapes on silicon to be produced. These simulators can account for several proximity effects, line edge roughness, etc. Some embodiments combine the data produced by these simulators with the process technology file information for the technology stack in order to produce the 3-D models with high precision. These 3-D models are then used as input to the field solver tools to extract capacitances with the high level of accuracy.

[0052] In cases where run-time is not as critical, the manufacturing simulation tools can fully take manufacturing process variations and curvilinear design techniques into account, allowing for a more accurate determination of curvilinear interconnect variability to be made across process corners. However, given that run-time is often critical, some embodiments use newer and better capacitance extraction techniques that do not depend on traditional pattern libraries and traditional pattern matching, while accounting for the impact of process variations, and the increasing presence of curvilinear shapes in manufactured designs. These embodiments enable accurate parasitic extraction for both curvilinear design shapes and manufactured curvilinear interconnect shapes while taking process variations into account.

[0053] Traditional capacitance extraction approaches depend primarily on CPU based processing, using Single Instruction, Single Data stream (SISD) processing architectures. While it is possible to divide the pre-characterization and extraction problems into region-based sub-problems that can be solved in parallel using a multiple CPU approach, the calculations for the sub-problems themselves are still not as fine-grained as the problems which are typically solved on Graphics Processing Units (GPUs) with Single Instruction, Multiple Data (SIMD) architectures – for example graphics processing or deep learning applications. Hence, for traditional capacitance extraction approaches, a huge number of CPUs are required to realize significant performance benefits.

[0054] It is therefore desirable to map the capacitance extraction problem onto new SIMD architectures such as GPU or Tensor Processing Unit (TPU) devices in order to obtain a far more fine-grained level of parallelism, and to be able to solve capacitance extraction

problems more efficiently. Some embodiments improve the speed of the parasitic pre-characterization and extraction by performing these operations in the pixel domain, which in turn allows for these operations to be performed by SIMD architectures such as GPU or Tensor Processing Unit (TPU) devices. These embodiments use machine trained networks (e.g., neural networks) to process the pixel-based definitions of the IC design components (e.g., wire structures) analyzed during the pre-characterization and extraction.

[0055] For instance, the systems and methods of some embodiments determine capacitance values using a field solver, in which the input conductor structures used as input to the solver are curvilinear (e.g., in the plan view). Some embodiments produce these curvilinear shapes using a trained curvilinear shape prediction convolutional neural network (CNN). The systems and methods of some embodiments perform a technology pre-characterization that trains a capacitance prediction CNN, and then stores the capacitance prediction network structure and trained weights in a pattern library. Some embodiments provide systems and methods that train a multiple track capacitance prediction CNN during technology pre-characterization and then store the capacitance prediction CNN structure and trained weights in a pattern library.

[0056] The use of a trained, curvilinear shape prediction CNN allows some embodiments to produce precise representations of 3-D manufactured curvilinear conductor shapes quickly at extraction time. These 3-D manufactured curvilinear conductor shapes are then provided as input to a field solver. This approach improves the accuracy of critical net extraction, and particularly the accuracy of critical net extraction in the presence of significant manufacturing process variations.

[0057] Some embodiments use a deep learning technique, instead of geometrical approaches, in order to perform capacitance coefficient modeling in pattern-based extraction for non-critical nets. For instance, in some embodiments, a capacitance component prediction CNN architecture is used to predict capacitance or capacitance coefficient values by using 2-D rasterized images of conductor structures as input, instead of geometrical parameters as input. As such, certain limitations of model-based or table-based approaches (such as used in 2.5D and 3-D pattern matching techniques) are removed. This, in turn, expands the applicability and range of pattern based techniques.

[0058] Some embodiments perform full capacitance extraction by using trained CNNs, instead of using traditional pattern matching or field solvers for parasitic extraction. For instance, in some embodiments, the design to be extracted is rasterized to the pixel domain, and split into image tiles. The capacitances for the conductors represented as pixels within each

tile are rapidly inferred by a trained capacitance prediction CNN and integrated over the tiles associated with a given conductor to obtain final capacitance values. The embodiments that use neural networks can be executed quickly as they can be efficiently processed by the SIMD underlying architectures of today's GPU and TPU devices.

[0059] **FIG. 6B** illustrates using a trained curvilinear shape-prediction neural network 650 in a capacitance extraction flow to produce highly accurate parasitic capacitance values for the curvilinear conductors that will result after using an IC design to manufacture an IC. The neural network 650 is trained to produce several 2-D curvilinear shapes 654 for several manufacturing process variations for 2-D shapes 652 that are defined after an EDA stage (e.g., after routing). In some embodiments, the neural network processes pixel definition of the 2-D shapes 652 and produces the curvilinear 2-D shapes 654 in the pixel domain. Accordingly, for the neural network, the input 2-D shapes 652 are rasterized into the pixel domain.

[0060] As shown in **FIG. 6B**, the produced 2-D curvilinear shapes 654 for the manufacturing process variations are then fed to the 3-D extrusion and meshing process 656, which then uses these shapes and information from the process technology file 658 to produce precise meshed 3-D shapes 660 (as defined by 3D surface descriptions). To perform the extrusion, the definition of the 2-D curvilinear shapes is transformed from the pixel domain to the geometric contour domain in which shapes are defined by the definition of their contours.

[0061] These 3-D shapes 660 are then provided as input to a field solver 662. The description of the 3-D shapes 660 is far more accurate than what would be produced using traditional methods. This, in turn, results in significantly more accurate capacitance values 364 being produced by the field solver 662.

[0062] Instead of just running one neural network 650 to produce several 2-D curvilinear shapes 654 for several manufacturing process variations, other embodiments use several single-output neural networks run in parallel, each for a different manufacturing process variation. These concurrently executed neural networks produce several process corner-specific 2-D wafer contours for several process variations. In some embodiments, each such neural network uses a pre-determined set of weights corresponding to one manufacturing process variation.

[0063] On the other hand, the neural network 650 that produces several 2-D curvilinear shapes 654 for several manufacturing process variations, takes as input a set of IC layout drawn shapes but produce not one but multiple outputs of curvilinear shapes, one per process manufacturing corner. Examples of multiple 2-D curvilinear shapes for multiple manufacturing process variations (produced by one neural network 650 or multiple single process-variation

networks) include a mean curvilinear image, a maximum curvilinear image, and a minimum curvilinear image, corresponding to different extremes in the processing conditions.

[0064] Details on these examples and on how curvilinear shape-prediction neural networks can be trained and subsequently used to produce detailed 2-D images of curvilinear silicon wafer shapes, given raster images derived from IC designs as input are disclosed in "Methods and Systems to Determine Shapes for Semiconductor or Flat Panel Display Fabrication," U.S. Application Publication 2022/0128899, and U.S. Provisional Application 63/283,520, filed November 28, 2021, both of which are incorporated herein by reference.

[0065] As mentioned above, the 2-D curvilinear shapes 654 over the process variations are extruded and meshed in parallel by the 3-D extrusion and meshing process 656, to form a set of corner-specific or extreme-specific 3-D surface meshed volumes 660, which are then input to the field solver 662. The field solver is just one field solver in some embodiments, while it is multiple field solvers in other embodiments. The field solving operation performed by the field solver(s) produces a corresponding set of parasitic capacitance values (e.g., a set of matrix values) over the manufacturing process variation. In some embodiments, the parasitic capacitance values are filtered and converted into DSPF/SPEF files (Detailed Standard Parasitic Format/Standard Parasitic Exchange Format), or other industry standard parasitic representations, such as the Synopsys Galaxy Parasitic Database (GPD).

[0066] The modifications to the curvature due to the various manufacturing process variation are thus accurately captured in the solver-produced capacitance values for the various process corners. For each 2-D shape in the IC design being analyzed, the above-described embodiments compute multiple 2-D curvilinear shapes over multiple process variations. However, other embodiments use the flow and neural network illustrated in **FIG 6B**, to generate 2-D and 3-D curvilinear shapes for just one process condition (one specific manufacturing process variation), and hence just produce parasitic capacitance values for this one process condition.

[0067] **FIG. 7** illustrates a novel non-geometrical approach 700 to replace or complement the geometrical approach for computing capacitance coefficients during a pre-characterization process that produces capacitance coefficients for later use during extraction. The geometrical approaches take a piece of layout and reduce it to geometric features, such as wire length, spacings, etc., as mentioned above by reference to **FIG. 4**.

[0068] On the other hand, during technology pre-characterization, the approach illustrated in **FIG. 7** replaces the simplistic capacitance models/lookup tables with a trained capacitance prediction neural network 740, which is a more universal function approximator.

Under this approach, image rasterization 720 is performed on a wire structure 710 to produce several 2-D images, which are referred to below as a multi-channel 2-D image 730. The image rasterization 720 defines the multi-channel 2-D image 730 in the pixel domain (i.e., produces a pixel-based definition for the multi-channel 2-D image 730). So, instead of quantifying geometrical attributes and then using these attributes along with pre-characterized lookup table values to compute capacitances, the approach illustrated in **FIG. 7** uses pixel representations of the design to produce capacitance coefficients from which parasitic capacitances are computed (e.g., after multiplying the coefficients by wire segment lengths and/or wire segment overlapping lengths).

[0069] In some embodiments, the image rasterization produces white pixels for fully-filled pixels (e.g., pixels fully covered by a shape, such as wire segments), black pixels for fully-empty pixels (e.g., pixels not covering any shapes, such as wire segments), and grey pixels for partially-filled pixels. In some of these embodiments, fully-filled pixels are represented with the numerical value 1.0, fully-empty pixels are represented as 0.0, and partially-filled pixels are represented with a value in the range [0,1] representative of the area of the pixel which is filled by the wire (e.g., a pixel that is 50% filled will have a value of 0.5). Before rasterizing the wire structure, some embodiments decompose the wire structure into several components (e.g., several wires, wire segments or wire structure portions), which it then individually rasterizes.

[0070] The multi-channel 2-D image 730 is then used as the primary input to the capacitance-predicting neural network 740, which produces a capacitance vector 750. In some embodiments, the capacitance vector values 750 produced by the trained capacitance neural network 740 are further post-processed into capacitance coefficients. To this end, the capacitance vector 750 is supplied to a post-processor 755, which produces capacitance coefficients 760 as output. As shown, these capacitance coefficients include Cl, Ca, Cf, Co, Cu in some embodiments, while in other embodiments they are post processed into other coefficients for other capacitance models.

[0071] To produce the capacitance coefficients 760, the post-processor 755 in some embodiments constructs simultaneous equations relating the capacitance matrix values to Cl, Ca and Cf capacitance component values, and solves this equations to produce Cl, Ca and Cf component values for that particular wire width and spacing, i.e., W, S pair. The approach is then repeated for various values of W, S. In some embodiments, the produced capacitance coefficients 760 are parasitic unit lengths. Hence, during extraction, the produced capacitance

coefficients 760 are then used to compute parasitic capacitances, for example, by multiplying these coefficients with length of overlapping wire segments.

[0072] Using the trained neural network 740 as the mapping mechanism is advantageous as during extraction phase, it removes the need for externally performed linear interpolation when wire widths and spacings are different from those used during training time. This is because neural networks, when appropriately designed and trained, act as universal function approximators, and removes the need for such external interpolation when operating on previously unseen data. Another advantage of this method over conventional approaches is that the multi-channel 2-D image 730 that is input into the neural network 740 may represent arbitrary conductor shapes, including curvilinear shapes.

[0073] For example, in some embodiments, one or more of the wire structures are generated from curvilinear shapes, by a second neural network that is trained for shape prediction (e.g., the curvilinear shape-prediction neural network of **FIG. 6B**). In some embodiments, the input semiconductor design includes several wire structures. Each of these wire structures is rasterized, and curvilinear shapes are calculated from each rasterized wire structure. The rasterizing of each wire structure can produce several images, e.g., with each image in some embodiments corresponding to one layer's of wiring in the wire structure. The produced curvilinear wire structures can include multiple wire tracks that cross each other, and can be sub-segmented into smaller structures. Before rasterizing each wire structure, some embodiments decompose the wire structure into several components (e.g., several wires, wire segments or wire structure portions), which it then individually rasterizes.

[0074] In **FIG. 7**, the neural network output (i.e., the capacitance vector 750) in some embodiments represents a vector of capacitance values for a geometric structure. Input wire structure 710 may be rasterized into multi-channel 2-D image 730 representing three conductors each on a layer i of interest, a crossover layer i+1 above, and a cross-under layer i-1 below. During rasterization 720, the wire structure is rasterized onto the multi-channel 2-D image 730. In some embodiments, the different image channels represent different interconnect layers of an IC manufacturing process.

[0075] For a 3x3 bus crossing structure, 9 nine capacitances are of interest as shown in **FIG. 16**, which are a self-capacitance for a central conductor segment 5 in a central layer i ("metal 2"), along with eight capacitance values to this segments eight nearby neighbors. These neighbors include (1) lateral capacitances from the central conductor to its left and right neighbor segments 4 and 6 on the central layer i, (2) three crossover capacitances from the central conductor segment 5 on the central layer i to three conductor segments 1, 2, and 3 on

layer i+1 (i.e., the layer above, "metal 3"), and (3) three cross-under capacitances from the central conductor segment 5 on the central layer i to three conductor segments 7, 8 and 9 on layer i-1 (i.e., the layer below, "metal 1").

[0076] To train the neural network 740, some embodiments use known input sets (e.g., known wire structures) with known output sets (e.g., known capacitance coefficients). To produce these known input/output sets, some embodiments use a field solver approach that will be described below by reference to **FIG. 9**. During training, groups of known input sets are rasterized and fed through the neural network 740 and post-processed (as shown in **FIG. 7**) to produce groups of output sets. The difference between each produced group of output sets and the known output sets of each group of known input sets is an error value that is propagated back through the neural network 740 to train its trainable parameters (e.g., its weight values). Some embodiments perform the training once per process technology, and then perform the operations of FIG. 7 to perform the extraction once or more than once during the IC design.

[0077] In other embodiments, the capacitance-prediction neural network may be trained to directly output the capacitance coefficient values themselves, e.g., the outputs may be capacitance coefficients (Cl, Ca, Cf, Co, Cu). With this approach, the post-processing step itself is also learned by the neural network. **FIG. 8** shows the architecture of a CNN 800 of some embodiments that can be used to directly output the capacitance coefficient values.

[0078] In this figure, a 3-channel input image 805 is processed by a convolutional base 810, that includes two pairs of convolution layers 815 and 820 (e.g., with 5x5 kernels each). Each of the convolution layers has a subsequent 2D max pooling 825 or 830 to down sample the images. Each convolutional layer uses a filter depth of 32. Input image dimensions are 60x60 pixels, with each pixel representing a 10 nm square of IC design data. Hence, each image represents a 600x600 nm area of the IC design.

[0079] In the neural network model, the convolutional base 810 is followed by a 16-neuron-wide fully-connected bottleneck layer 835, which serves to reduce the overall number of model parameters. The output from this narrow layer is then fed to a regression network 840, which includes a 100-neuron-wide fully-connected layer followed by a 9-neuron fully-connected output layer.

[0080] All layers with the exception of the final output layer use ReLU activation, and all convolutional layers use zero-padding to ensure the output image size is the same as the input image size. Since this is a regression CNN application, the final output layer uses a linear activation function. The final output layer is as wide as the number of capacitances N, to be predicted. Once training is complete, a set of trained weights for each process technology in

some embodiments is preserved for use in a neural network, such as capacitance-prediction neural network 740. In other embodiments, the final output layer is as wide as the number of capacitance coefficients to be predicted. One of ordinary skill in the art will understand that other embodiments use neural network structures different than the structure illustrated in **FIG. 8** to produce parasitic values.

[0081] To train the CNN 800 or another neural network to produce parasitic values, some embodiments use training data set with known inputs and output values. These embodiments iteratively (1) feed sets of known input values successively through to the neural network to produce sets of output values, (2) compute an error value between each set of produced output values and the known output values corresponding to the input values, and (3) back propagate each computer error value through the neural network in order to adjust the configurable parameters of the neural network (e.g., its weight values) to reflect the knowledge gained through the training.

[0082] **FIG. 9** represents a data flow diagram for creating the training data set (X and Y data) necessary for training a neural network, e.g., a 3x3 bus crossing structure. The X training data represents a rasterized wire structure input, while Y training data represents a capacitance vector output. In this example, several 3-layer, 2-D rectangular wire crossing structures 900 are generated with different wire widths and spacings. Some embodiments use different wire widths down to a minimum of 30 nm, with a unit wire length of 90 nm. Spacings of up to 4 routing tracks wide are also used, as spacings beyond this are commonly assumed to lead to relatively inconsequential changes in capacitance values.

[0083] An image rasterizer 910 performs a rasterization operation on each wiring structure 900 to produce a 2-D image that is defined in the pixel domain for the wiring structure. Each 2-D image has 3 channels each containing a 2-D rasterized image representing a layer i with a first preferred routing direction (e.g., vertical), and the layers above and below with orthogonal second preferred routing directions (e.g., horizontal). In some embodiment, a pixel size of 10 nm is used during rasterization so that, for example, a 30 nm-wide wire is rendered as 3 pixels wide in the image. Fully-filled pixels are represented with the numerical value 1.0, fully-empty pixels are represented as 0.0, and partially-filled pixels are represented with a value in the range [0,1] representative of the area of the pixel which is filled by the wire (e.g., a pixel that is 50% filled will have a value of 0.5).

[0084] Each wire-crossing structure rasterized in this manner for input to the neural network is then fed to an extrusion and mesh modeling process 920, which produces a 3-D representation of the structure. As mentioned above, to perform the extrusion operation that

uses the contour definition of shapes, some embodiments convert the definition of the 2-D curvilinear shapes from the pixel domain to the geometric contour domain in which shapes are defined by the definition of their contours.

[0085] The produced 3-D representations are suitable for input to a field solver 925. To create the field solver input representation, the extrusion and mesh modeling process 920 uses the 2-D wire dimensions from the wire structure with the various layer-specific wire heights and dielectric thickness as specified in the process technology file 915 for the manufacturing process. This allows the 2-D wire shapes to be extruded in the 'height' dimension forming 3-D volumes. The set of process technology information in the process technology file can include wire heights and dielectric information, for example.

[0086] For each of the resultant 3-D interconnect volumes, some embodiments compute the surface panels. In some embodiments, these panels are simply computed as rectangles. In more complex embodiments, these panels are computed by applying a more complex meshing algorithm before extrusion, for example as described above with respect to computing curvilinear 3-D interconnect shapes. For instance, some embodiments produce triangular or quadrilateral meshes. Ground planes are then inserted above and below the top and bottom layers.

[0087] The 3-D surface panel representations, including added ground planes above and below the layer of interest, are then solved by the field solver 925, producing an NxN capacitance matrix, where N is the total number of conductors. For a 3x3 bus crossing structure, there is a total of 9 conductors, and so the field solver will produce a 9x9 matrix with 81 capacitance values. A filter 930 then filters down these values to just the primary capacitance component values of interest, e.g., the self-capacitance of the central layer, central conductor, and the capacitances between that conductor and each of its 8 neighbors, as shown in **FIG. 16**.

[0088] For each candidate geometric wire structure, the training data generation flow of **FIG. 9** generates sample wire structure using a range of different widths and/or spacings. Each generated wire structure is a known input X. To produce this input's corresponding known output Y, the training data generation flow (1) produces the 3-channel rasterized image for each generated wire structure, (2) produces this image's extruded 3-D representation, and then (3) generates the filtered capacitance vector output by the field solver and filter. The filtered capacitance vector is the known output Y of the training set with the known input X. The large set of samples so generated is partitioned into a training set (e.g., 80% of the samples), and a validation set (e.g., 20% of the samples), following deep learning best practices.

[0089] **FIG. 10** illustrates a data flow diagram for a CNN-based full capacitance extraction method. An IC design 1002 containing 2-D layout shapes on multiple layouts has its shapes rasterized by an image rasterizer 1003. The rasterized images (defined in the pixel-domain) are then provided as input to a trained curvilinear prediction neural network 1004 running on GPU/TPU devices (a single corner-specific set of curvilinear shapes in the figure for simplicity). The neural network 1004 performs a rapid inference operation that produces a set of process-corner-specific curvilinear 2-D shapes representing what will be manufactured on a substrate at each process corner.

[0090] The curvilinear prediction network may be trained using the methods disclosed in the above-incorporated U.S. Application Publication 2022/0128899. The resulting wafer shape contours are computed and stored in a database 1006. For each corner-specific set of curvilinear wafer shapes stored in the database 1006, the corresponding curvilinear interconnect wire segments of a net to be extracted are located via shape-chasing process 1008. As shown, this process also uses the originally drawn wafer shapes in the IC layout and their corresponding connectivity.

[0091] The process 1008 also breaks the curvilinear wire segments into sub-segments, each representing a 2T+1 track-width length of interconnect wire with T being the number of tracks. In some embodiments, the number of tracks T equals 4 but other embodiments use a different number of tracks (e.g., 5). Each sub-segment is then explored in the X and Y directions in a square region to find the nearest neighbor wires on the same layer, and the crossover/crossunder wires on the interconnect layers above and below, within +/- T tracks of the interconnect of interest.

[0092] In some embodiments, the process 1008 produces a square tile 3-channel raster image 1012 to represent each interconnect sub-segment, its same-layer nearest neighbors within +/- T tracks to the left and right, and up to 2T+1 crossover/crossunder wires in the vicinity on the layers above and below. The capacitance array to be inferred will contain 2*(2T+1)+3 slots (e.g., 21 slots, for 4 tracks). Each such sub-segment tile is rendered as a 3-channel, two-dimensional image. In other embodiments, the tile image for the middle layer *i* capture up to T lateral neighbors on either side of the conductor being extracted, i.e., not just the two closest lateral neighbors. In this case, the capacitance array to be inferred will contain 3*(2T + 1) slots (e.g., 27 slots, with 9 per each of 3 layers when 4 tracks are used).

[0093] The generated sub-segment tiles representing the interconnect sub-segments are then passed into a trained capacitance prediction neural network 1014, which predicts/infers coupling capacitance values. In some embodiments, the neural network 1014 has a similar

architecture to the neural network 800 of **FIG. 8**, though the number of outputs is different (for example, 23 or 27 outputs when 4 tracks are used). The neural network 1014 computes all relevant capacitances for each sub-segment tile (also called sub-segment area below). In sum, **FIG. 10** includes a first neural network 1004 to produce predicted curvilinear shapes from the rasterized images of the IC design shapes, and a second neural network 1014 to calculate predicted parasitic capacitances of the curvilinear shapes using the set of process technology information.

[0094] As shown, the output of the second neural network 1014 are the sub-segment, tile specific capacitances 1016 that are supplied as input to an integration process 1018. For each interconnect segment from the original layout, the process 1018 gathers the segment's related sub-segment-specific capacitances and integrates over all the related sub-segment tiles. This flow adds the computed capacitances together according to the interconnect connectivity. This integration will be further described below by reference to **FIG. 11**. The process 1018 outputs the integrated capacitance values for all interconnects to a standard parasitic format 1020, such as DSPF/SPEF files, or Synopsys GPD files for example.

[0095] **FIG. 11** provides an example to illustrate the tiling and rasterization process used in some embodiments. This example illustrates the manufactured curvilinear shapes of some conductors of interest. In this example, the method extracts the capacitances with respect to a central vertical conductor 1102 on layer "1/0" in the design. Intuitively, the conductor 1102 will have large lateral capacitances to its same layer neighbors 1104 and 1106 to the left and right, which run parallel to it substantially though with different spacings. The conductor 1102 will also have fringing/overlap capacitances to shapes on layers above (layer "2/0" in the design) and layers below (layer "0/0" in the design).

[0096] Thus, the tiling process of some embodiments tiles the vertical conductor 1102 into multiple sub-segments. In this example, the tiling process results in three tiles 1112, 1114 and 1116, each containing a portion of the center vertical conductor. Each tile also contains two same-layer nearest neighbor lateral conductor sub-segments. Additionally, three conductor shapes within a 4-track window are also present on the top and bottom orthogonally routed layers.

[0097] Each 3-layer tile from the tiling process is then rasterized into a 3-channel, 2-D 60x60 pixel raster image 1120 with 10 nm pixels. **FIG. 11** illustrates the 3-channel image for the tile 1112. Here, the 3-channel image is broken out separately, such that each channel image 1122-1126 shows pixels representative of the interconnect sub-segments on the respective interconnect layer.

[0098] During inference, the 3-channel 2-D tile raster images are input to the capacitance prediction convolutional neural network 1014, and up to 27 capacitance values are predicted for each of the 3 tiles, representing the self-capacitance of the central layer center conductor, its coupling capacitance to the conductors in the lateral neighboring tracks, and its coupling capacitance to each of the crossover/crossunder tracks represented in the top/bottom channels of the tile image. As mentioned above, for the example described in **FIG. 10**, quantities other than 27 may be used in other embodiments.

[0099] After inferencing, the resulting capacitances are then summed over all tiles, i.e., are added together according to the connectivity of the interconnect portions. For example, to obtain the total per-corner self-capacitance of the central vertical conductor 1102 on layer "1/0", the self-capacitances of the central conductor across each of the three tiles 1112-6 are summed.

[00100] **FIG. 12** depicts a flow to generate training data with which to train the neural network for tile-based capacitance extraction. In this example, a variety of multi-layer (e.g., 3-layer), 2-D interconnect structures are generated that fit in squares comprising NxN, such as 9x9, track slots. The structures are stored in an IC design database 1202 and split out into their individual layers. An image rasterizer 1204 then rasterizes the structure content for each layer as a single-channel, 2-D image, and then combines the individual layers across three layers into a 3-channel, 2-D image.

[00101] These raster images are then consumed by the curvilinear prediction neural network 1206. An example of such a neural network is the neural network described above or one of the neural networks disclosed in U.S. Application Publication 2022/0128899. This neural network 1206 outputs per process corner curvilinear images 1210 corresponding to the outputs of the manufacturing process. The curvilinear 2-D wafer shapes for the three interconnect layers are gathered for each process corner and supplied to an extrusion process 1212.

[00102] This process 1212 extrudes the curvilinear 2-D wafer shapes to 3-D using the layer-specific wire height and dielectric information in the process technology file 1214, and the resulting 3-D interconnect structures are input to the field solver 1216. The field solver produces an NxN capacitance matrix 1218 per process corner as output, which is then filtered by the filter 1222 to the capacitances of interest, i.e., the coupling capacitance between the center layer central conductor of interest, and the conductors on the other tracks on all three layers.

[00103] For each interconnect structure sample, the per-corner 3-channel 2-D curvilinear image tile used as output from the curvilinear shape-prediction neural network 1206, and the corresponding per-corner capacitance vector output by the filter 1222 are gathered as the inputs X and outputs Y respectively to be used when training the capacitance prediction neural network.

[00104] During the generation of the 2-D 3-layer structures, each structure may contain wire segments of various lengths, appearing in any of the valid track positions. Wires may run the full width or height of the tile, or may run for a partial width or length. In some embodiments, wires are placed on the available track positions, using a set range of wire lengths. For any track position except that of the central layer center conductor, the wire length may be as short as zero length, i.e., wire may be absent in a particular track location. Non-zero wire lengths may be quite short (e.g., one track width), or may run the full 9-track width of the structure. The start/end position of each wire within its track may also be snapped to a set range of positions, e.g., the routing track crossing points.

[00105] Some embodiments allow the training space to be sampled in a structured, grid-like manner. In other embodiments, a Monte Carlo approach is taken to populate the training space. Here, wire start and end positions for each wire are randomly generated for each track. Again, wire lengths for any track position (except for the center layer central conductor) may be as short as zero. Training samples with empty track positions are assigned a capacitance value of 0 at those positions, and solver-produced capacitance values are used for the non-empty positions.

[00106] **FIG. 13** illustrates the training data sets used in some embodiments to train a neural network that produces parasitic capacitance values. As shown, these embodiments use curvilinear wire shapes as the input values of each training set and the known capacitance values associated with these curvilinear wire shapes as the output values of each training set. Training samples using set of process technology information produces a set of trained weights which are preserved for each process technology.

[00107] **FIG. 14** illustrates an example of a computing hardware device 1400 that may be used to perform the calculations described in this disclosure. Computing hardware device 1400 comprises a central processing unit (CPU) 1402, with attached main memory 1404. The CPU may comprise, for example, eight processing cores, thereby enhancing performance of any parts of the computer software that are multi-threaded. The size of main memory 1404 may be, for example, 64 G-bytes. The CPU 1402 is connected to a Peripheral Component Interconnect Express (PCIe) bus 1420. A graphics processing unit (GPU) 1414 is also

connected to the PCIe bus. In computing hardware device 1400 the GPU 1414 may or may not be connected to a graphics output device such as a video monitor. If not connected to a graphics output device, GPU 1414 may be used purely as a high-speed parallel computation engine. The computing software may obtain significantly higher performance by using the GPU for a portion of the calculations, compared to using CPU 1402 for all the calculations. The CPU 1402 communicates with the GPU 1414 via PCIe bus 1420. In other embodiments (not illustrated) GPU 1414 may be integrated with CPU 1402, rather than being connected to PCIe bus 1420. Disk controller 1408 may also be attached to the PCIe bus, with, for example, two disks 1410 connected to disk controller 1408. Finally, a local area network (LAN) controller 1412 may also be attached to the PCIe bus, and provides Gigabit Ethernet (GbE) connectivity to other computers. In some embodiments, the computer software and/or the design data are stored on disks 1410. In other embodiments, either the computer programs or the design data or both the computer programs and the design data may be accessed from other computers or file serving hardware via the GbE Ethernet.

[00108] **FIG. 15** is another embodiment of a system for performing the computations of the present embodiments. The system 1500 may also be referred to as a CDP, and includes a master node 1510, an optional viewing node 1520, an optional network file system 1530, and a GPU-enabled computing node 1540. Viewing node 1520 may not exist or instead have only one node, or may have other numbers of nodes. GPU-enabled computing node 1540 can include one or more GPU-enabled nodes forming a cluster. Each GPU-enabled computing node 1540 may comprise, for example, a GPU, a CPU, a paired GPU and CPU, multiple GPUs for a CPU, or other combinations of GPUs and CPUs. The GPU and/or CPU may be on a single chip, such as a GPU chip having a CPU that is accelerated by the GPU on that chip, or a CPU chip having a GPU that accelerates the CPU. A GPU may be substituted by another co-processor.

[00109] The master node 1510 and viewing node 1520 may be connected to network file system 1530 and GPU-enabled computing nodes 1540 via switches and high-speed networks such as networks 1550, 1552 and 1554. In an example embodiment, networks 1550 can be a 56 Gbps network, 1552 can be a 1 Gbps network and 1554 can be a management network. In various embodiments, fewer or greater numbers of these networks may be present, and there may be various combinations of types of networks such as high and low speeds. The master node 1510 controls the CDP 1500. Outside systems can connect to the master node 1510 from an external network 1560. In some embodiments, a job may be launched from an outside system. The data for the job is loaded onto the network file system 1530 prior to launching the

job, and a program is used to dispatch and monitor tasks on the GPU-enabled computing nodes 1540. The progress of the job may be seen via a graphical interface, such as the viewing node 1520, or by a user on the master node 1510. The task is executed on the CPU using a script which runs the appropriate executables on the CPU. The executables connect to the GPUs, run various compute tasks, and then disconnect from the GPU. The master node 1510 may also be used to disable any failing GPU-enabled computing nodes 1540 and then operate as though that node did not exist.

[00110] While the specification has been described in detail with respect to specific embodiments, it will be appreciated that those skilled in the art, upon attaining an understanding of the foregoing, may readily conceive of alterations to, variations of, and equivalents to these embodiments. These and other modifications and variations to the present methods may be practiced by those of ordinary skill in the art, without departing from the scope of the present subject matter, which is more particularly set forth in the appended claims. For instance, even though curvilinear shapes are mentioned as being used by some embodiments, one of ordinary skill will realize that rectilinear or arbitrary shapes are used to represent a design in other embodiments.

[00111] Furthermore, those of ordinary skill in the art will appreciate that the descriptions above are by way of example only, and is not intended to be limiting. Steps can be added to, taken from or modified from the steps in this specification without deviating from the scope of the invention. In general, any flowcharts presented are only intended to indicate one possible sequence of basic operations to achieve a function, and many variations are possible. Thus, it is intended that the present subject matter covers such modifications and variations as come within the scope of the appended claims and their equivalents.

CLAIMS

What is claimed is:

1. A method for calculating parasitic parameters for a wire structure that is to be manufactured on a substrate, the method comprising:

receiving the wire structure;

performing a rasterization operation on the wire structure to produce a pixel-based definition for each of a plurality of images that collectively represent the wire structure; and

using the pixel-based definition of the plurality of images as inputs to a neural network to calculate parasitic parameters of the wire structure.

2. The method of claim 1, wherein the neural network accounts for variations in process technology.

3. The method of claim 2, wherein the neural network uses a pre-determined set of weights corresponding to each variation in process technology.

4. The method of claim 1, wherein each image in the plurality of images correspond to a layer in the wire structure.

5. The method of claim 1, wherein the wire structure comprises a plurality of rectilinear shapes.

6. The method of claim 1, wherein the wire structure comprises a plurality of curvilinear shapes.

7. The method of claim 6, wherein the neural network is a first neural network, and the curvilinear shapes are generated by a second neural network that is trained for shape prediction.

8. The method of claim 7, wherein the curvilinear shapes are generated from multiple wire tracks that cross each other.

9. The method of claim 8, wherein the generating further comprises sub-segmentation of the curvilinear shapes.

10. The method of claim 1, wherein the parasitic parameters include at least one parasitic capacitance value representing predicted, unwanted parasitic capacitance exerted on at least one wire of the wire structure.

11. The method of claim 10, wherein the parasitic parameters include a plurality of parasitic capacitance values representing a plurality of predicted, unwanted parasitic capacitances exerted on a plurality of wires of the wire structure.

12. A non-transitory machine readable medium storing a program for calculating parasitic parameters for a wire structure that is to be manufactured on a substrate, the program comprising sets of instructions for:

receiving the wire structure;

performing a rasterization operation on the wire structure to produce a pixel-based definition for each of a plurality of images that collectively represent the wire structure; and

using the pixel-based definition of the plurality of images as inputs to a neural network to calculate parasitic parameters of the wire structure.

13. The non-transitory machine readable medium of claim 12, wherein the neural network accounts for variations in process technology.

14. The non-transitory machine readable medium of claim 13, wherein the neural network uses a pre-determined set of weights corresponding to each variation in process technology.

15. The non-transitory machine readable medium of claim 12, wherein each image in the plurality of images correspond to a layer in the wire structure.

16. The non-transitory machine readable medium of claim 12, wherein the wire structure comprises a plurality of rectilinear shapes.

17. The non-transitory machine readable medium of claim 12, wherein the wire structure comprises a plurality of curvilinear shapes.

18. A method for training a first neural network to extract parasitic capacitance from a semiconductor design, the method comprising:

receiving a semiconductor design comprising a plurality of wire structures;

rasterizing the plurality of wire structures;

calculating curvilinear shapes from the rasterized plurality of wire structures;

receiving a set of process technology information;

calculating parasitic capacitances of each of the curvilinear shapes, by using the set of process technology information, wherein the calculating produces a set of trained weights for the set of process technology information; and

outputting the set of trained weights for the set of process technology information.

19. The method of claim 12, wherein the calculating of curvilinear shapes is performed by a second neural network.

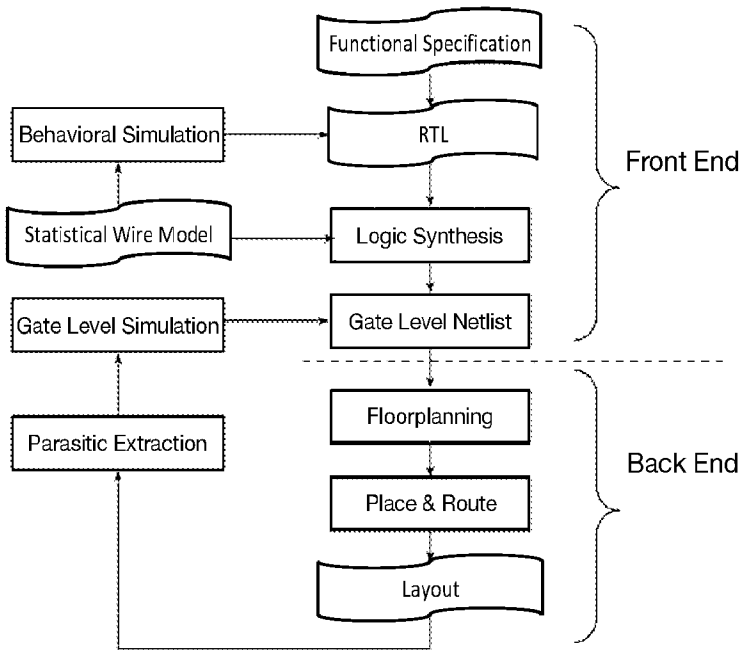

FIG. 1
(Prior Art)

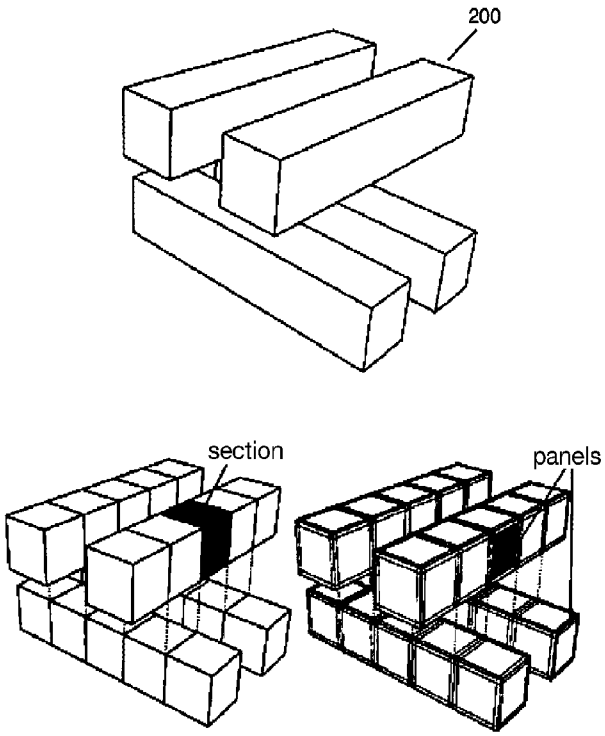

FIG. 2
(Prior Art)

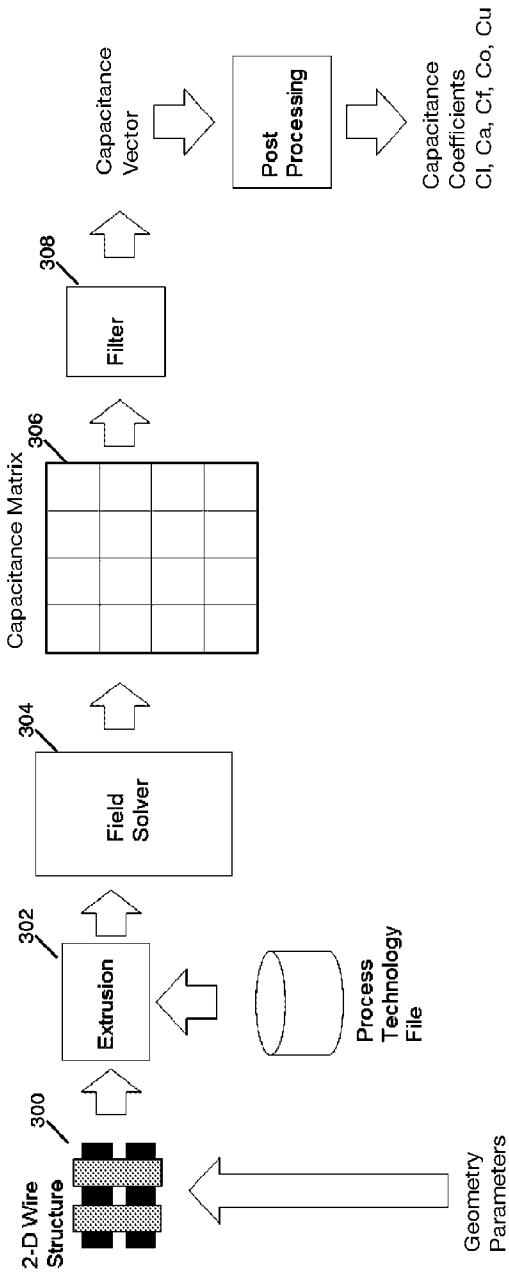

FIG. 3
(Prior Art)

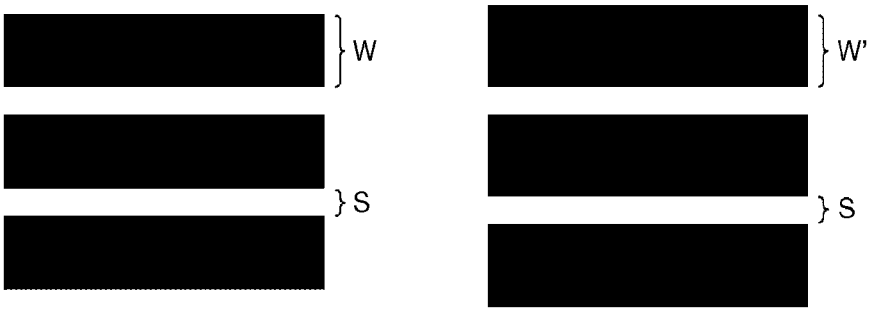

FIG. 4A
(Prior Art)

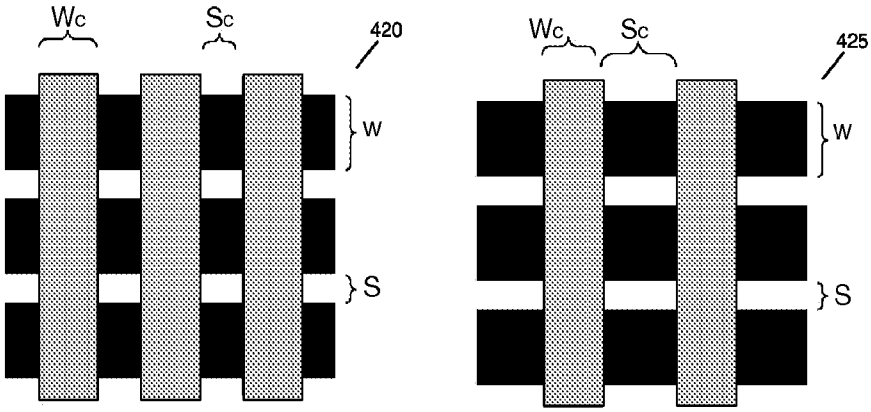

FIG. 4B
(Prior Art)

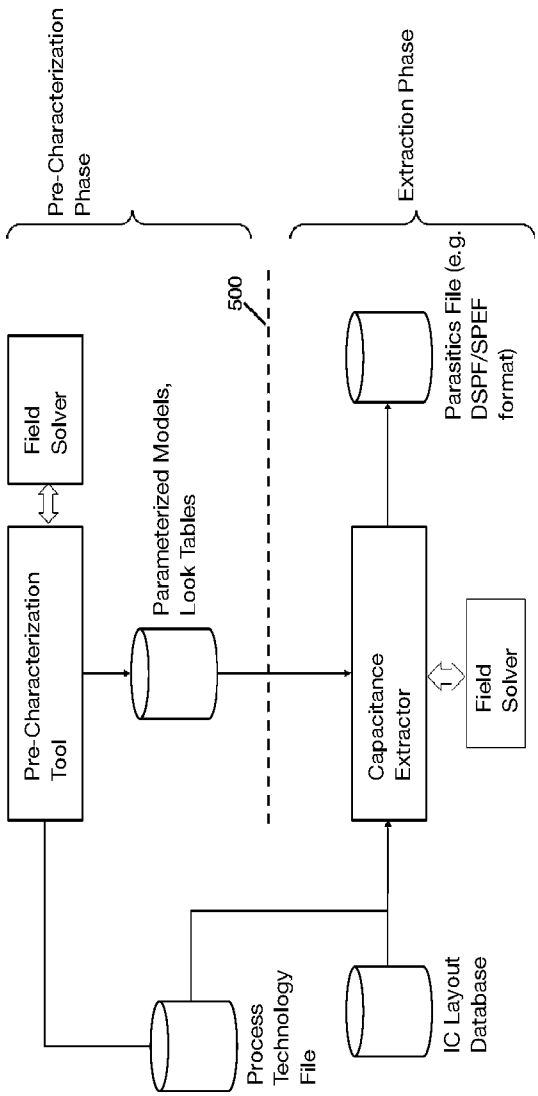

FIG. 5
(Prior Art)

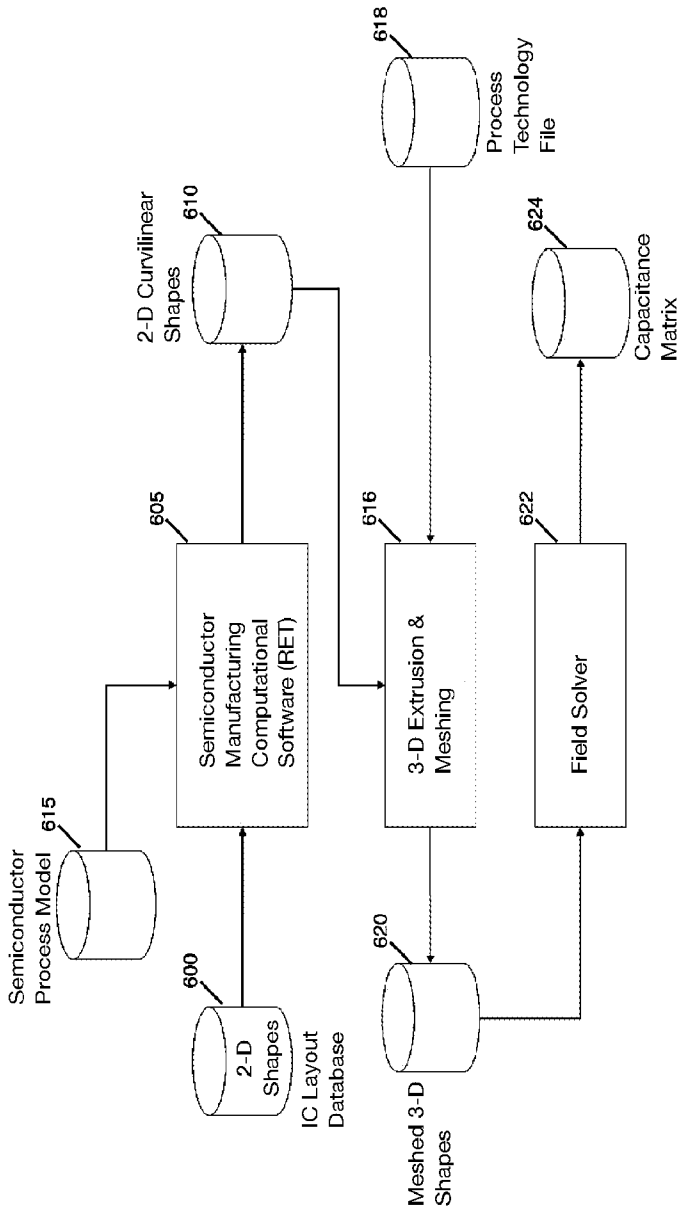
618
Process Technology File
610
2-D Curvilinear Shapes
624
Capacitance Matrix
605
Semiconductor Manufacturing Computational Software (RET)
616
3-D Extrusion & Meshing
622
Field Solver
615
Semiconductor Process Model
600
2-D Shapes
IC Layout Database
620
Meshed 3-D Shapes

FIG. 6A

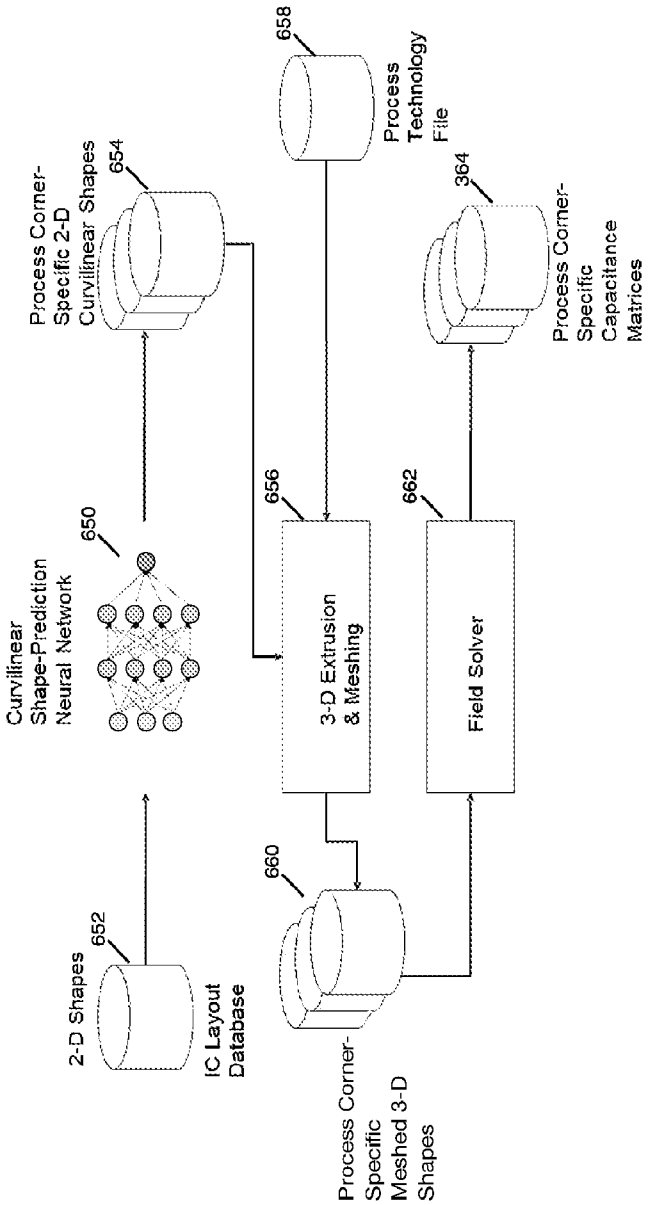
658
Process Technology File
654
Process Corner-Specific 2-D Curvilinear Shapes
364
Process Corner-Specific Capacitance Matrices
650
Curvilinear Shape-Prediction Neural Network
656
3-D Extrusion & Meshing
662
Field Solver
660
652
2-D Shapes
IC Layout Database
Process Corner-Specific Meshed 3-D Shapes

FIG. 6B

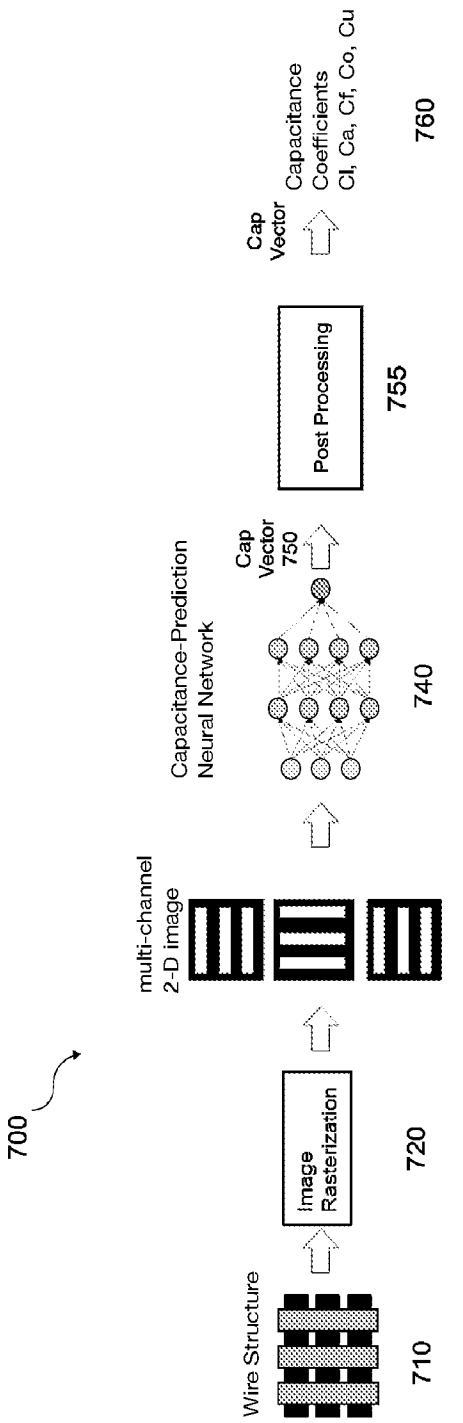
700
Wire Structure
710
Image Rasterization
720
multi-channel 2-D image
730
Capacitance-Prediction Neural Network
740
Cap Vector 750
Post Processing
755
Cap Vector
Capacitance Coefficients
Cl, Ca, Cf, Co, Cu
760

FIG. 7

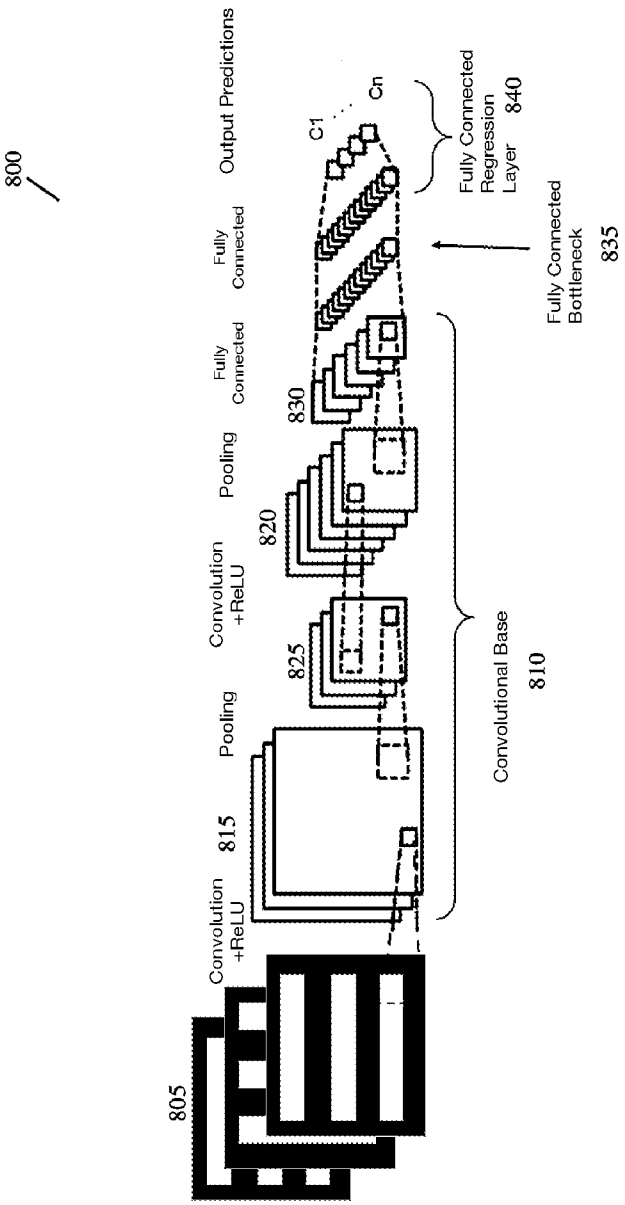
800
805
Convolution +ReLU
815
Pooling
825
Convolution +ReLU
Pooling
820
Fully Connected
830
Fully Connected
Output Predictions
C1
Cn
Fully Connected Regression Layer
840
Fully Connected Bottleneck
835
Convolutional Base
810

FIG. 8

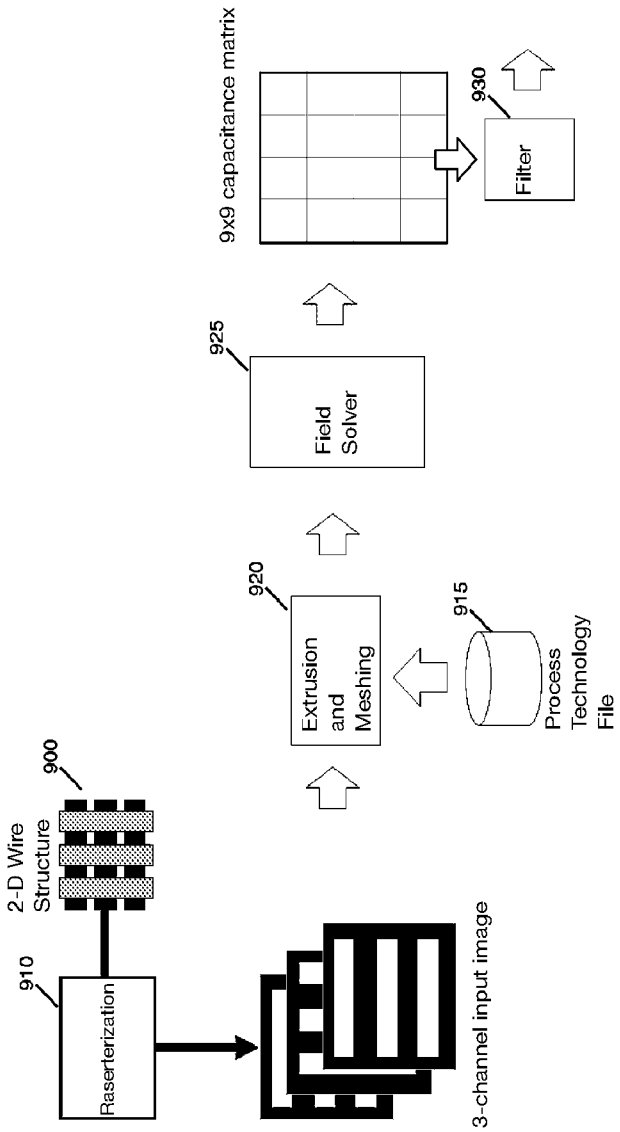
900
2-D Wire Structure
910
Raserterization
3-channel input image
920
Extrusion and Meshing
915
Process Technology File
925
Field Solver
9x9 capacitance matrix
930
Filter

FIG. 9

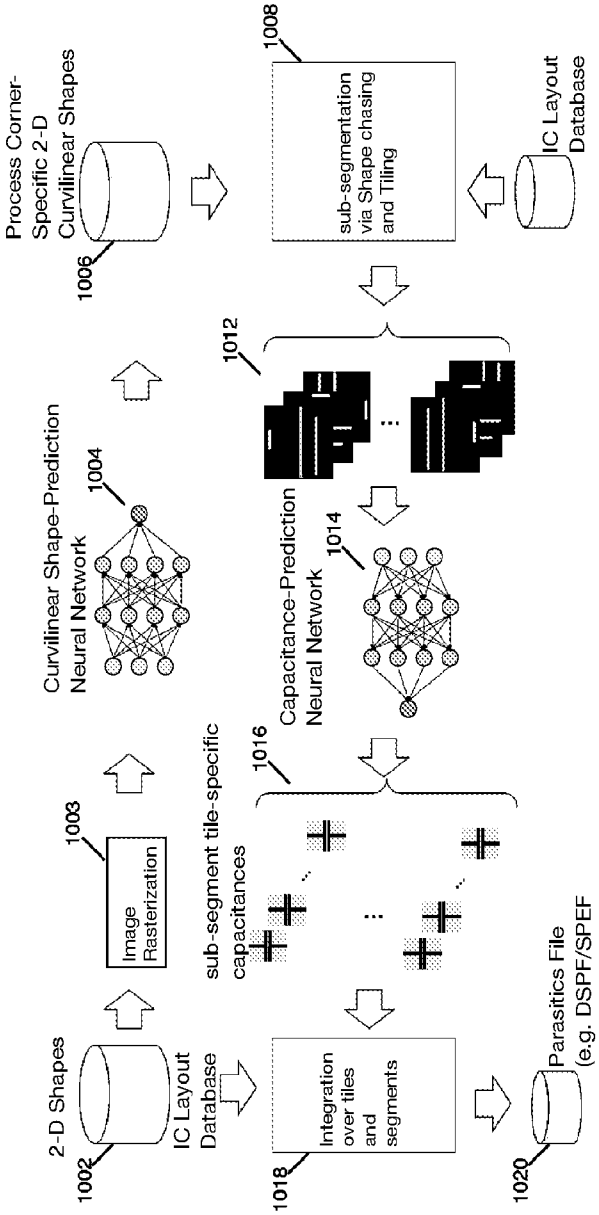
2-D Shapes
IC Layout Database
1002
Image Rasterization
1003
Curvilinear Shape-Prediction Neural Network
1004
Process Corner-Specific 2-D Curvilinear Shapes
1006
sub-segmentation via Shape chasing and Tiling
1008
IC Layout Database
1012
Capacitance-Prediction Neural Network
1014
sub-segment tile-specific capacitances
1016
Integration over tiles and segments
1018
Parasitics File (e.g. DSPF/SPEF format)
1020

FIG. 10

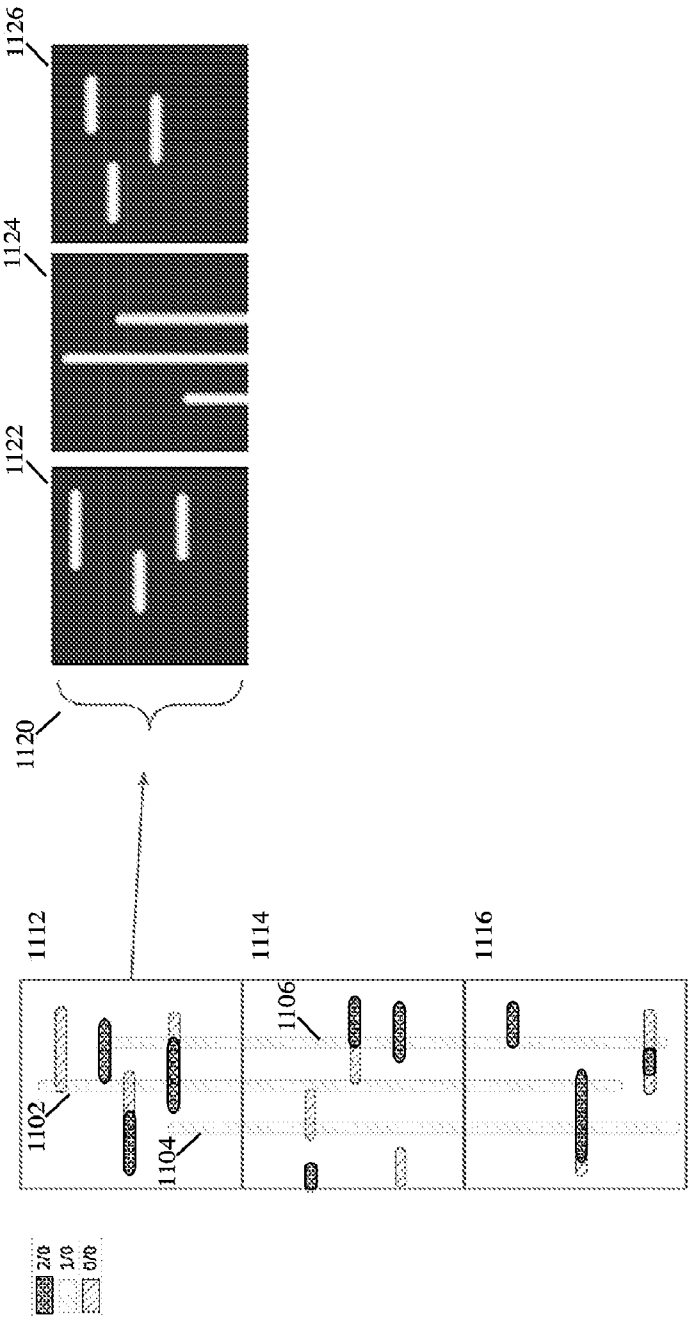

FIG. 11

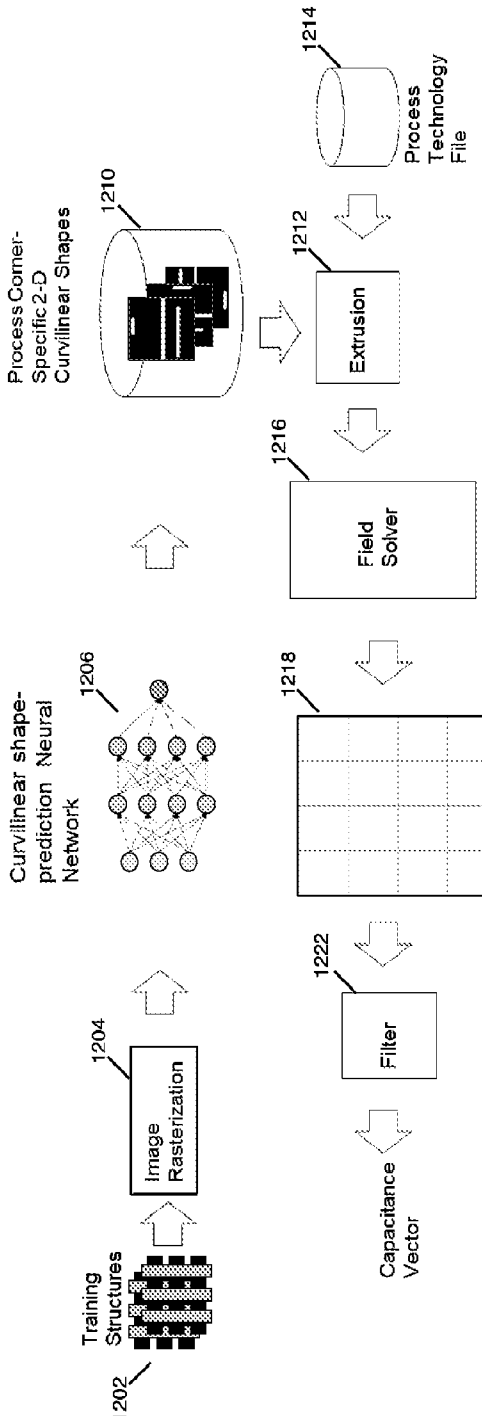
1214
Process Technology File
1210
Process Corner-Specific 2-D Curvilinear Shapes
1212
Extrusion
1216
Field Solver
1206
Curvilinear shape-prediction Neural Network
1218
1222
Filter
1204
Image Rasterization
Training Structures
1202
Capacitance Vector

FIG. 12

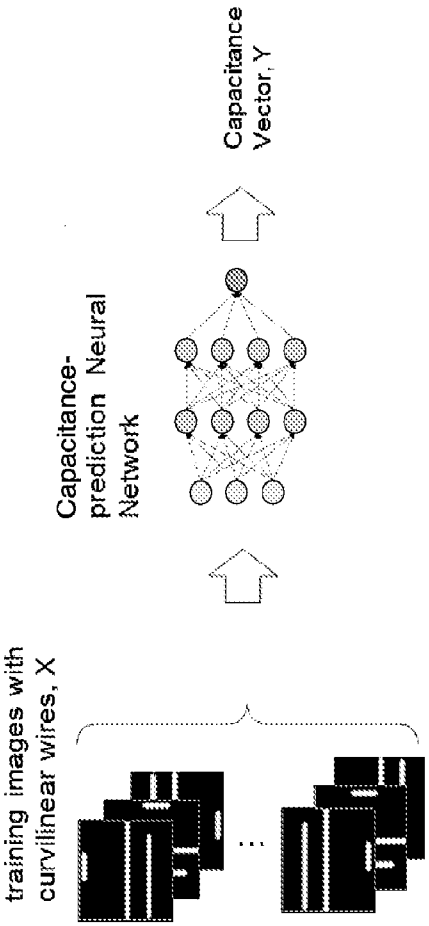
training images with curvilinear wires, X
Capacitance-prediction Neural Network
Capacitance Vector, Y

FIG. 13

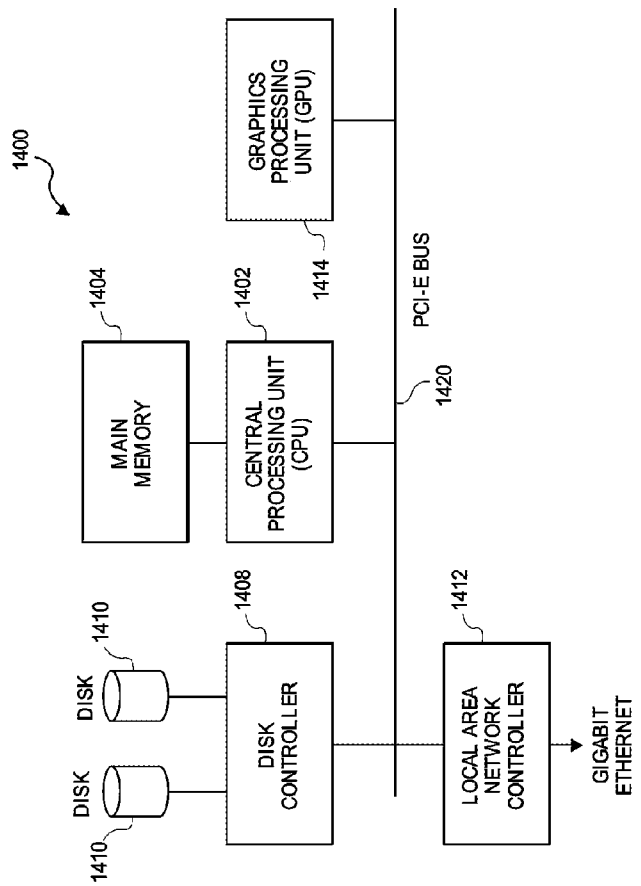
1400
GRAPHICS PROCESSING UNIT (GPU)
1414
1404
1402
MAIN MEMORY
CENTRAL PROCESSING UNIT (CPU)
PCI-E BUS
1420
1408
1410
DISK
DISK
1410
DISK CONTROLLER
1412
LOCAL AREA NETWORK CONTROLLER
GIGABIT ETHERNET

FIG. 14

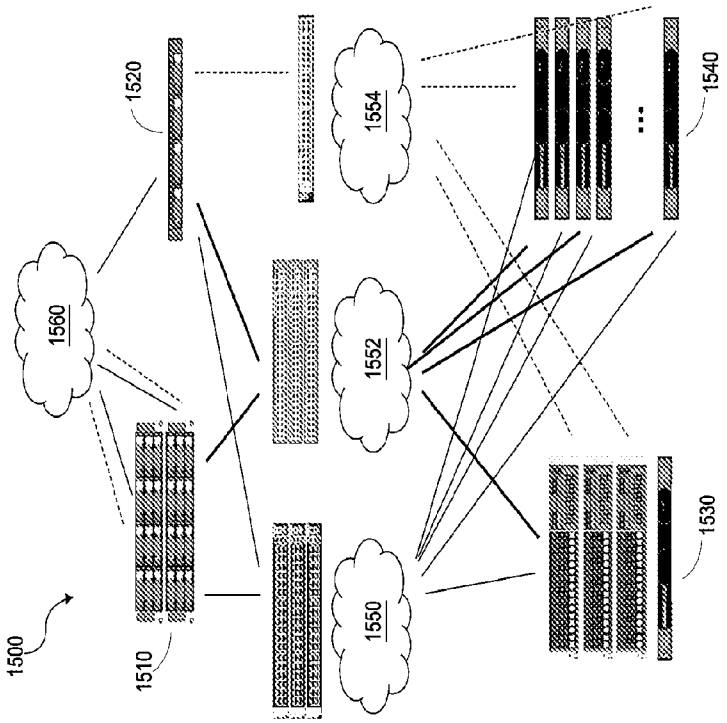

FIG. 15

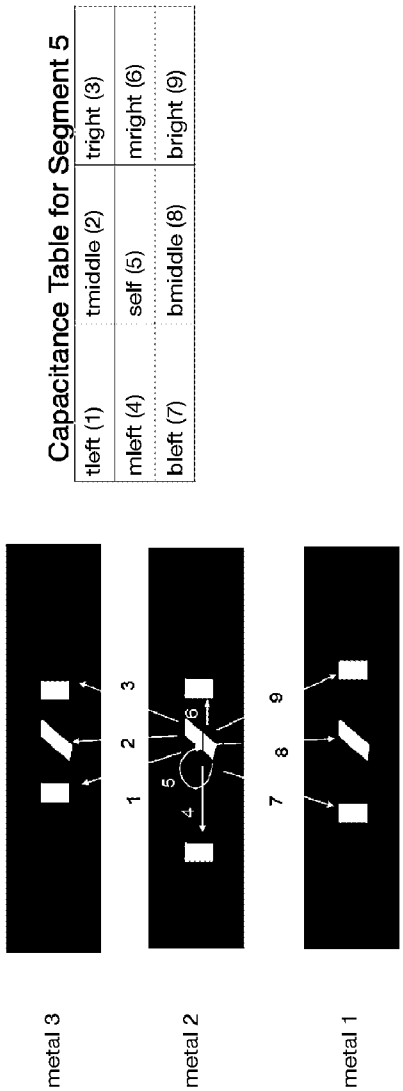
Capacitance Table for Segment 5
tleft (1)
tmiddle (2)
tright (3)
mleft (4)
self (5)
mright (6)
bleft (7)
bmiddle (8)
bright (9)
metal 3
metal 2
metal 1
1
2
3
4
5
6
7
8
9

FIG. 16