Moonwalk: NRE Optimization in ASIC Clouds or, *accelerators will use old silicon*

Moein Khazraee, Lu Zhang, Luis Vega, and Michael Bedford Taylor
UC San Diego

Abstract

Cloud services are becoming increasingly globalized and data-center workloads are expanding exponentially. GPU and FPGA-based clouds have illustrated improvements in power and performance by accelerating compute-intensive workloads. ASIC-based clouds are a promising way to optimize the Total Cost of Ownership (TCO) of a given datacenter computation (e.g. YouTube transcoding) by reducing both energy consumption and marginal computation cost.

The feasibility of an ASIC Cloud for a particular application is directly gated by the ability to manage the Non-Recurring Engineering (NRE) costs of designing and fabricating the ASIC, so that it is significantly lower (e.g. 2×) than the TCO of the best available alternative.

In this paper, we show that technology node selection is a major tool for managing ASIC Cloud NRE, and allows the designer to trade off an accelerator's excess energy efficiency and cost performance for lower total cost. We explore NRE and cross-technology optimization of ASIC Clouds for four different applications: Bitcoin mining, YouTube-style video transcoding, Litecoin, and Deep Learning. We address these challenges and show large reductions in the NRE, potentially enabling ASIC Clouds to address a wider variety of datacenter workloads. Our results suggest that advanced nodes like 16nm will lead to sub-optimal TCO for many workloads, and that use of older nodes like 65nm can enable a greater diversity of ASIC Clouds.

Keywords NRE, ASIC Cloud, TCO, datacenter, accelerator

1. Introduction

With the impending end of CMOS scaling and Moore's Law, the research community has increasingly looked towards designing ASIC-based accelerators that exploit specialization in order to surpass power- and energy- limited general-purpose devices like CPUs and GPUs. ASIC accelerators are able to attain order-of-magnitude improvements in energy-efficiency (W per op/s) and cost-performance ($ per op/s) over general-purpose substrates.

Although research in accelerators has been widespread, translation of these accelerators into commercial practice has proven challenging for two key reasons: *deployment friction* and *Non-Recurring Engineering (NRE)* costs. In this introduction, we discuss recent trends that have reduced deployment friction, and then in the rest of the paper, examine the second challenge, NRE minimization, that is, the minimization of all costs required to create and deploy an ASIC accelerator.

We employ the term deployment friction to refer to the difficulty of deploying these accelerator designs into a real-world computing ecosystem. For accelerators that target client devices, deployment of a researcher's accelerator often requires convincing Apple, Intel, or Qualcomm to add the accelerator to their high-volume SoCs, a difficult technology transfer problem with complex social and economic aspects. Beyond the standard organizational barriers, these companies must be convinced that customers will pay extra money to provide sufficient additional profit over the increased cost across a large number of price-sensitive parts. In many cases, emerging applications may not have achieved sufficiently wide-spread use to make 1-accelerator-to-1-device deployment economically appropriate, eliminating the incentive to place the accelerator on the die. Moreover, the customer may not use the application enough to create a perceivable benefit in terms of battery life or productivity.

The Cloud Reduces Deployment Friction. The cloud, on the other hand, provides intriguing possibilities for deployment of discrete accelerators. Because software and hardware are vertically integrated in many cloud contexts, companies like Google, Facebook, Microsoft, Apple, and Amazon can custom-design their hardware–from server to PCB to chip–for recurring workloads that impart significant total-cost-of-ownership (TCO) to their business units.

As the Cloud/mobile bifurcation of computation continues, we see growing classes of planet-scale workloads (think Facebook's face recognition of uploaded pictures, or Apple's Siri

ASPLOS '17, April 08-12, 2017, Xi'an, China

DOI: http://dx.doi.org/10.1145/3037697.3037749

voice recognition, or the IRS performing tax audits with neural nets) where datacenters are performing the same computation across many users. This kind of scale-out workload can easily leverage discrete chips that contain NoC-connected arrays of replicated compute accelerators (RCAs), and provides the economic incentive that covers the NREs of ASIC development. Moreover, unlike in the client side, the resources spent on the accelerator can be scaled with the workload by adding more racks, whereas accelerator to non-compute ratio in a mobile phone population is set at tapeout.

Our previous papers on *ASIC Clouds* [39] examined the construction of datacenters of ASIC accelerators for Bitcoin, Litecoin, Video Transcoding (e.g. Youtube), and Neural Networks. Subsequently, Google announced the existence of their Tensor Processing Unit (TPU) [1], which is used to implement a neural network ASIC Cloud. Bitcoin and Litecoin ASIC Clouds already exist, and the economics behind transcoding clouds are very strong. These ASIC Clouds vastly improve upon both the energy efficiency and marginal cost of performing the computation by orders of magnitude. These two metrics are the primary TCO drivers in the datacenter.

Because of both the reduction in deployment friction and the potential TCO improvements, Doug Carmean of Microsoft Research in his ISCA 2016 keynote referred to the datacenter as "the architect's new playground" – essentially expressing excitement at his realization that there is exponentially less deployment friction in the context of the datacenter and that the upsides are large (and indeed, Microsoft is examining FPGA-based datacenters, DNA-storage and even Quantum machine learning). For the remainder of this paper, we examine a key remaining challenge in deploying accelerators into this "architect's playground": NRE.

The two-for-two rule for NRE and TCO. The high up-front costs of developing an ASIC system are collectively known as non-recurring engineering costs (NRE). These NRE costs for ASIC accelerators are comprised of labor, tool, IP, and mask costs. Ultimately the deployment of accelerators in the clouds must result in a net financial benefit for the underlying company. How can we relate the performance and energy efficiency benefits of an accelerator to the NRE and the (pre-accelerated) TCO of the targeted workload? Our recent paper [39] proposed the *two-for-two rule*, which states an accelerator-based cloud at least breaks even when both of these two conditions hold:

1. **The computation's TCO $>2\times$ the NRE.** So for example, if YouTube spends $30 million a year on video transcoding, and the NRE of development the accelerator is $10 million, a $3\times$ ratio, they clearly pass the bar.
2. **The TCO per op/s benefit of the ASIC Cloud $>2\times$.** So for example, if YouTube's transcoding accelerator has, all-inclusive, at least 2X improvement in energy efficiency, and at least 2X benefit in performance per $ of hardware, they must have at least a 2X benefit in TCO per op/s, and would also fulfill these criteria. In [39], the authors compute an improvement in excess of $10{,}000\times$ in TCO per op/s for a 28nm video transcoding ASIC Cloud versus an Intel Xeon server, trivially meeting this criteria.

As we can see, the TCO per op/s benefit of the 28nm transcoding ASIC Cloud easily clears the $2\times$ bar. In fact, surveying the academic literature, most accelerators easily clear this bar by several orders of magnitude. *Thus, the major gating factor for ASIC Cloud deployment is NRE, not the performance of the accelerator.*

The rest of the paper. Accordingly in this paper, we explore ways to reduce NRE, and in particular look at ways to **trade the excess of performance and energy efficiency of accelerators for reductions in NRE cost**. We examine the implications of using older technology nodes, and also quantify differences in NRE across different applications.

The contributions of this paper are:

1. We build a model for the NRE costs of developing ASIC accelerators.
2. We show that process technology node (e.g. 16nm) is one of the most important knobs for controlling the NRE cost.
3. We show that targeting older nodes reduces both mask costs and IP costs exponentially, but only moderately changes labor and tool costs.
4. We show how to analyze an accelerator and determine the optimal technology node based on the TCO of the computation.
5. We show that advanced nodes like 16nm are optimal only for a limited set of ASIC Clouds, and that nodes like 180nm, 65nm and 40nm can broaden the applicability of ASIC Clouds. We introduce the concept of a *tech parity node* which can, when combined with pre-ASIC datacenter TCO, be used to estimate the ideal target node for an application.

2. Understanding Silicon Tradeoffs in Technology Node Selection

In this section, we analyze the tradeoffs of available technology nodes, with the goal of allowing ASIC servers to select across technology nodes to find the TCO+NRE optimal designs. Our rich menu of available nodes means that we have an equally rich tradeoff space that links *mask cost* NREs, *energy efficiency* (i.e. joules per operation), *cost efficiency* (i.e. $ per op/s, a function of frequency, transistor count and wafer cost), *maximum transistor count* per accelerator, and *frequency* (i.e. serial performance per accelerator).

Unlike in client ASIC designs, for scale-out ASIC Cloud servers, the die area, power budget and performance of each individual chip is not critical, so long as the workload's latency, throughput and cost requirements are met. By examining the TCO of the target computation, we can optimize across process nodes and correctly weight the importance of

cost-efficiency, energy-efficiency, and NRE to attain cost savings in an ASIC Cloud workload. The maximum transistors per die metric filters those nodes that do not have sufficient transistor density to fit even a single accelerator. Similarly, the transistor frequency metric serves to filter out process nodes that do not offer the required single-accelerator speed or latency.

Overview. Figure 1 examines the different metrics, based on data we collected from four sources in order of preference: 1) using CAD tools in our lab, 2) via Internet disclosures of technical data 3) by interviewing industry experts, and 4) using CMOS scaling to interpolate missing points. Recall that in CMOS scaling, the factor S refers to the ratio of feature widths of two nodes; for example, given 180nm and 130nm, S=180/130=1.38×. Typical scaling factors between successive nodes are often assumed to be S=1.4×. Typically, transistor count increases with S^2, transistor frequency with S, and transistor capacitance (and energy per op at a fixed voltage) decreases with S.

Because of our use of historical and current data rather than predictive scaling theory, our nodes are different than typical scaling theory nodes, reflecting the reality of available process technology. In today's nodes, 40nm has supplanted 45nm, and 28nm has supplanted 32nm. We exclude 20nm because it has been supplanted with 16nm FinFET.

Although most of the tech node feature widths are spaced by S=1.4×, 65nm and 40nm are spaced by S=1.6×, and 28nm and 16nm are spaced by S=1.75×. Accordingly, we have plotted the data on a log-log plot with the X axis plotting feature width. Thus a straight line with slope of 1 indicates feature-width-proportional scaling. For mask costs, we have standardized on 9 metal layers if the process supports it, and otherwise the maximum number of layers for older processes (i.e. 5 layers for 250nm and 6 layers for 180nm). More metal layers entails more masks, incurring more NRE.

Due to the nature of CMOS scaling, these metrics improve exponentially with more advanced process nodes. At the same time, mask NRE worsens exponentially as nodes advance. The space from 250nm to 16nm spans a **89× range in mask cost**, a **152× range in energy/op**, a **28× range in cost per op/s (558× for non-power density limited designs)**, a **256× range in maximum accelerator size** in transistors, and a **15.5× range in maximum transistor frequency**. Note that the Y axis typically spans two decades of range, but frequency is only slightly more than one decade, and transistor count spans a full three-decades.

Mask Costs. Figure 1-A and Table 1 show mask costs, which range from ∼65K for 250nm to almost ∼6M for 16nm. Mask cost scaling with feature width actually varies widely, as indicated by the varying slope of the segments. For example, 65nm and 40nm are particularly cheap steps, and 180nm to 130nm is a large step, relative to the previous node. Overall, mask cost multiples are smaller after 90nm than before, possibly because the number of metal layers has stabilized.

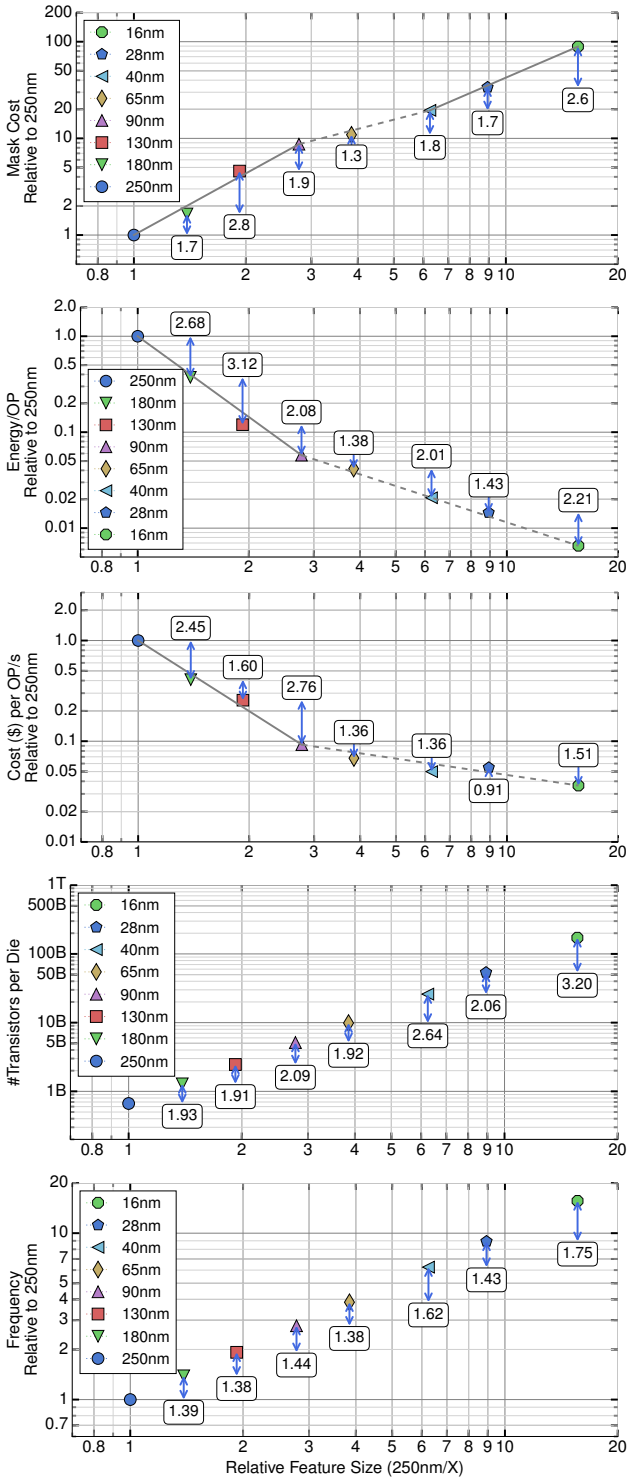


Figure 1: **Node Technology trade-offs, normalized to 250nm.** #'s indicate multiplicative benefits as node advance. Lines in mask cost indicate different regimes of mask cosk scaling. The dotted lines in energy and cost per op/s graphs indicate the post-Dennard slowdown in voltage scaling.

Tech	250nm	180nm	130nm	90nm	65nm	40nm	28nm	16nm
Mask cost ($)	65K	105K	290K	560K	700K	1.25M	2.25M	5.70M
Cost per wafer ($)	720	790	2,950	3,200	3,300	4,850	7,600	11,100
Wafer diameter (mm)	200	200	300	300	300	300	300	300
Backend labor cost per gate ($) [30]	0.127	0.127	0.127	0.127	0.127	0.129	0.131	0.263

Table 1: **Wafer and mask costs rise exponentially with process node.** Backend cost per gate jumps with double-patterning.

Energy per Op. As can be seen in Figure 1-B, energy per op (e.g. CV^2) improvements are markedly different after 90nm. This coincides with the end of Dennard scaling [55] after 90nm. Prior to 90nm, energy improvements were driven by S voltage scaling and by S capacitance scaling, and in 65nm and later, they are driven by S capacitance scaling and only marginal voltage scaling (about 1.04× per node, post-Dennard scaling, as shown in Table 2). *Thus, given that energy per op is a major TCO driver, the benefits of nodes after 90nm are much more limited than before 90*, when selecting a node for our ASIC Cloud. As a result, the penalty for going to a lower-NRE process is lower than might be implied by the node feature size.

Marginal cost per op/s. Figure 1-C graphs $ per op/s, i.e. the marginal silicon cost of adding computing capacity in a throughput-dominated workload such as typical in scale-out cloud applications. $ per op/s is hurt by exponentially increasing wafer costs, but helped by improvements in wafer size and ops/mm^2 compute density due to transistor frequency and density scaling. Table 1 shows that wafer costs scale approximately with S, but are also related to wafer size. During Dennard scaling, compute density improves as S^3, but below 90nm, it is limited to S by power density. 28nm has higher $ per op/s than 40nm because wafer cost rises faster than usable compute density improves. For applications that are not power-limited in 90nm, scaling continues more as a straight-line continuation of the pre-90nm curve, but bends towards the power-limited case at advanced nodes. In the results section of this paper, many of the accelerators operate the logic at below-nominal Vdd levels (e.g 0.5–0.8V), in order to improve performance within the thermal budgets.

Maximum design size. Figure 1-D graphs the maximum number of logic transistors per die; memories are scaling less well than shown in this graph. Generally speaking, transistors per die mostly places limits on how old a process node can be used before the accelerator does not fit.

Transistor Frequency. Transistor frequency improvements are graphed in Figure 1-E. For post-Dennard nodes that are power-density limited and do not operate at maximal clock rates, this metric still tracks the frequency of SerDes in DRAM controllers and high-speed off-chip interfaces. At older nodes, frequency limits accelerator serial performance, potentially resulting in unsatisfied datacenter latency or Service Level Agreement (SLA) requirements.

Tech Node (nm)	250	180	130	90	65	40	28	16
Nom. V_{dd} (V)	2.5	1.8	1.2	1.0	1.0	0.9	0.9	0.8

Table 2: **Real nominal supply voltages for each tech node.**

2.1 Takeaways

In this section, we examined the fundamental properties of the nodes available for use in creating tradeoffs between NRE and accelerator metrics. We saw that using a range of technology nodes provides a wide dynamic range of potential accelerator implementations. Nodes newer than 90nm show reduced marginal benefit in terms of the energy efficiency of the accelerator. These nodes also show reduced marginal cost benefits per unit of accelerator computation, because of post-Dennard scaling and rising wafer costs. Finally, mask costs for 65nm and 40nm are particularly cheap given their feature widths. The next section examines other NRE drivers.

3. ASIC Cloud: Architectural Overview

We give a quick high-level overview of an ASIC Cloud, shown from right to left in Figure 2; a more complete version is in [39]. The goal of an ASIC Cloud is to scale out a collection of application-specialized *replicated compute accelerators* (RCAs) in order to minimize both energy and capital components of TCO. RCAs are replicated inside an ASIC chip and interconnected with an interconnection network to an off-chip router. The flip-chip packaged ASICs are arranged in rows called *lanes* on a PCB, and interconnected to each other via an on-PCB network, as well as to the off-PCB interface. Each lane is enclosed by a duct and has a dedicated fan blowing air through it across the ASIC heatsinks. The PCB, fans and power supply are enclosed in a 1U server, which is then assembled into racks in a datacenter. Typically an FPGA or microcontroller serves as a bridge scheduling remote procedure calls (RPCs) that come from the off-PCB interface (1-100 GigE, RDMA, PCI-e, etc) on to the ASICs. The PCB is customized around the needs of the particular accelerator, including per-ASIC DRAM or HyperTransport links. Based on ASIC needs, the PSU and DC/DC converters are customized for each server.

4. Building a model for NRE

Previously, we examined mask costs, which can be a significant component of NRE, totaling as much as 90% of the NRE in advanced-node Bitcoin and Litecoin designs examined in

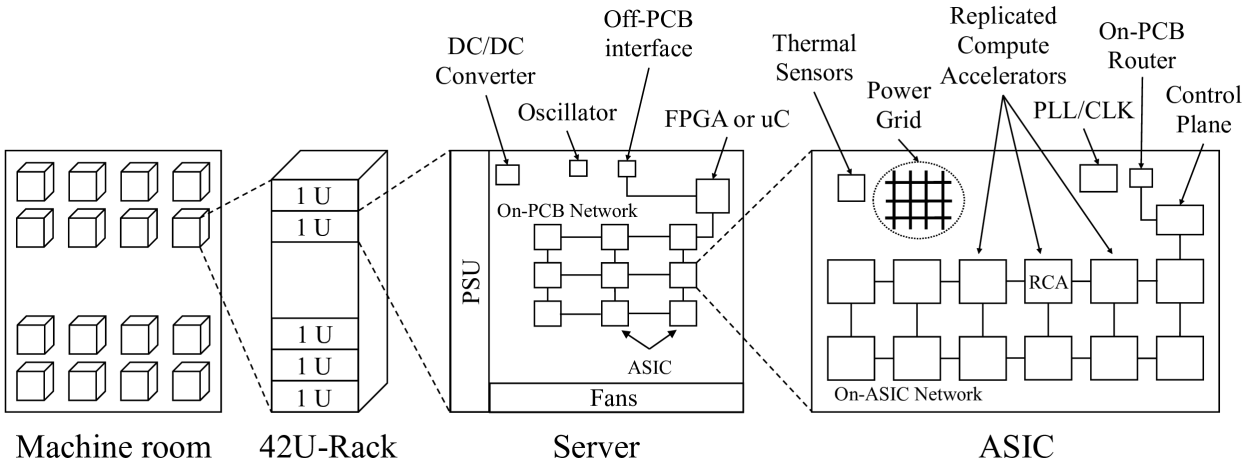


Figure 2: **High-Level Abstract Architecture of an ASIC Cloud [39].**

this paper. However, other NRE components can be significant. On old tech nodes with many third-party IP blocks, these non-mask NRE's can total up to 95% of the cost. In this section, we describe our model that incorporates principle components for NRE in ASIC development: labor, package design, CAD tool, IP, and mask costs.

Note that silicon wafer cost and per package cost are not NRE costs. These costs are part of the marginal cost and they are paid per ASIC. Later, in this paper, we will incorporate both NRE and marginal costs in order to determine Pareto- and NRE- optimal systems.

Packaging costs. Flipchip package design and tooling costs contribute about $105K to NRE, shown in Table 3.

Labor costs. Labor costs include application-to-architecture design time, frontend development (e.g. Verilog) and testing costs, backend design and verification costs (known as Verilog-to-GDS), IP validation costs (the significant cost of adopting somebody else's IP) and non-ASIC costs like PCB design, system-level interface development, and Cloud API coding. ASIC frontend design mainly involves *IP qualification*, *design specification*, *RTL implementation*, *module integration* and *functional testing*. The backend process consumes human time in *floorplanning*, *power and clock networking*, *placement*, *routing*, *timing closure*, *design rule verification*, and a few more marginal tasks before signing-off the chip layout. System NRE includes the system level code development for interfacing the ASICs with outside world, including firmware for the server's FPGA controller, FPGA code for job distribution across ASICs, and software modifications to an existing cloud to use ASIC Cloud servers. Also, ASIC Cloud servers require a custom PCB design.

Based on our analysis, frontend labor costs do not vary much with technology node, and relate more to design complexity (as measured imperfectly in lines of code, functional blocks, or gates.) For backend labor costs, costs scale with the number of unique design gates being mapped to the die, and by the complexity of the target node. Advanced nodes like 16nm that employ double-patterning suffer an additional multiplier based on greatly escalated back-end design costs. Since the ASIC Clouds employ regular arrays of accelerators on-die connected by a simple NoC (Network on Chip), we assume a hierarchical backend CAD flow that scales with RCA complexity rather than raw instance count on the die. A fixed gate count overhead is considered for I/O and NoC at the top-level of the chip. Frontend and backend labor salary rates as well as top-level overheads are shown in Table 3. 65% overhead is assumed for employee benefits and supplies.

Tool Costs. The tool costs include the frontend tools (e.g. Verilog Simulation and Synthesis), backend tools (e.g. RTL-to-GDS tools like Synopsys IC Compiler or Cadence Innovus), and PCB design tools. Of these tools, the backend tools are by far the most expensive. The model described in [30] gives the total backend labor cost in terms of gates. To calculate the required man-months for backend CAD tools, we divide the backend cost by the backend labor salary.

IP Costs. Each application's IP licensing cost depends on that application's specific IP requirements. Almost all accelerators will need standard cells (e.g. VLSI layouts for the gates, and basic LVCMOS I/O cells) and generator programs for making SRAMs. Typically, these are provided free for nodes at 65nm and older, and cost $100K or so for advanced nodes at 40nm & up. Designs that use fast ($>$ 150 MHz) clocks need an internal PLL. For systems that use DRAM, two IP blocks are required: a DRAM controller, and a DRAM PHY, the mixed-signal block that does high-performance signaling outside the chip. Similarly, for high-speed interfaces like PCI-E or HyperTransport, a controller and PHY IP block are required. Simple applications like Bitcoin may not need any IP beyond the standard cells, while a video transcoder might require a DRAM PHY, and a neural network ASIC Cloud might require a PCI-E or HyperTransport block. These IP costs greatly escalate the NRE of these accelerators. Table 4 shows typical IP licensing costs.

IP Cost Correlation with Nodes. In our investigation illustrated by Figure 3, **we have found that IP costs rise rapidly as the technology node increases**, and that the most expen-

Frontend Labor Salary [19]	$/yr	115K
Frontend CAD Licenses	$/Mm	4K
Backend Labor Salary [19]	$/yr	95K
Backend CAD Licenses	$/month	20K
Overhead on Salary		65%
Top-level gates		15K
NRE, flip-chip BGA package	$	105K

Table 3: **Node-independent NRE parameters in San Diego, CA in late 2016.** Mm=man-month. Backend Tools are more expensive than the people using them. Flip-chip packages add significant NRE.

sive IP blocks in general are PHY blocks found in PCI-E and DDRs. For 180nm and 250nm, no DDR DRAM blocks are available, and so a free SDR controller suffices. At advanced nodes like 16nm PCI-E and DDR cost almost $1M.

Having detailed our model for NRE, we now proceed by introducing our ASIC Design flow methodology, which takes a Verilog description of an accelerator (or metrics from a research paper's evaluation of an accelerator in a particular node), and a process node and then generates families of TCO-optimized ASIC Cloud servers.

5. ASIC Cloud Design Methodology

In order to evaluate design trade-offs and in choosing technology node, we model our investigation using the methodology proposed by Magaki et al [39], which provides a flow for going from a Verilog specification of an accelerator to a full server-level design including PCB design, assembling accelerators into ASIC chips, packaging, power and cooling system design using Computational Fluid Dynamics (CFD) simulation. We review the basic server parameters in the next section, and then discuss its adaptation for NRE optimization. We employ the same benchmark applications as used in that work, and the same datacenter TCO model [8].

Tech Node (nm)	250	180	130	90	65	40	28	16
DRAM Ctlr	NA	NA	125	125	125	125	125	125
DRAM PHY	NA	NA	150	165	175	280	390	750
PCI-E Ctlr	NA	NA	90	90	125	125	125	125
PCI-E PHY	NA	NA	160	180	325	375	510	775
PLL	15	15	15	20	30	50	35	50
LVDS IO	7.5	7.5	0	150	90	36	40	200
Standard Cells, SRAM	0	0	0	0	0	100	100	100

Table 4: **IP Licensing Costs increase with advancing Technology Nodes.** Commonly used IP licensing costs across tech nodes, in late 2016, thousands of USD. Costs generally rise with node, but there are some irregularities.

Application	Bit-coin	Lite-coin	Video Tr-anscode	Deep L-earning
RCA gate count	323K	96.7K	3.56M	1.51M
FE CAD-months	8	12	23	26
FE Mm	9.5	15	24	30
FPGA job distr. code, Mm	1	1	3	2
FPGA "BIOS" code, Mm	1	1	1	1
Cloud Software, Mm	2	2	7	6
PCB Design cost ($)	37K	37K	50K	37K

Table 5: Application-dependent NRE parameters. PCB design costs are for late 2016. Mm = man-months, FE = Front End

5.1 Server Parameters

Using the server architecture described in Section 3, we explore the design space for different numbers of RCAs per die, different number of ASICs per lane, and different logic voltages. If DRAM is required, we explore different numbers of DRAM memory controllers and DRAMs per ASIC. The limiting constraints in this space exploration are maximum junction temperature limits, maximum die size, maximum number of dies per lane, and finally voltage range based on fabrication technology. For each selection of die size and number of dies per lane, the optimal heatsink is selected by optimizing fin count and thickness as well as base thickness.

5.2 Optimal Server Design

To choose the optimal design, we use TCO analysis which incorporates server cost, server power and also datacenter-level constraints such as power delivery, land and interest into account. To compare different design points, server TCO is divided by its performance to find the optimal TCO per op/s. Cost per performance and power per performance (W per op/s, i.e., J per op) are the main factors for server evaluation.

We explain the trade-off between these two main components. Utilizing larger dies amortizes the fixed part of server cost and energy among more RCAs and also increases the portion of server cost designated to working accelerators; however, it lowers the maximum viable power density for the chip due to junction temperature limit. Assuming fixed total silicon per lane, increasing the number of chips alleviates this problem but increases the cost of packaging. As total silicon

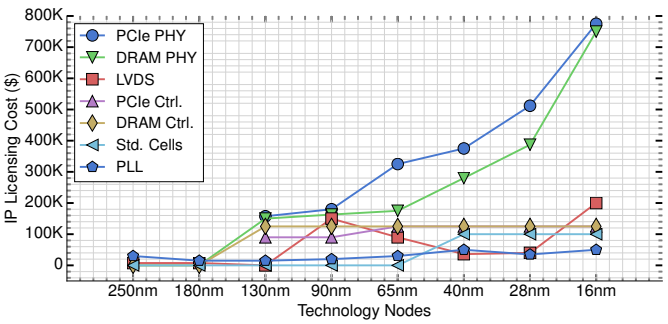


Figure 3: **IP Licensing Costs increase with advancing Tech Nodes.** High-speed I/O blocks rise exponentially.

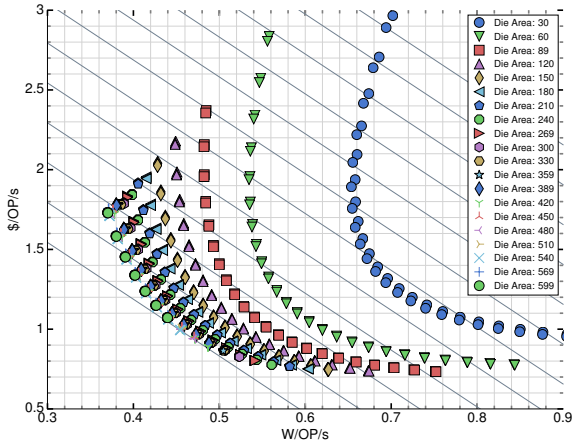


Figure 4: **Pareto curve example for Bitcoin application in 28nm technology.** Exploring different Die Area and logic voltage for optimal TCO per performance point. Voltage increases from left to right. Diagonal lines show equal TCO per performance values and the closer to the origin the lower the TCO per performance. This plot is for 9 ASICs per lane.

per lane increases, the amount of heat that can be extracted from each die is degraded and hence there is a tradeoff between number of dies per lane and silicon area of each die. On the other hand, increases in logic voltage translates to better use of same silicon and improvement on performance, based on the voltage-delay curve of the fabrication technology, but increases the power density and lowers the power efficiency.

Considering these factors, lowering the logic voltage increases the energy efficiency and more silicon can be used, resulting in minimal W per op/s or Joules/op, but high silicon cost and hence high $ per op/s. On the other side, increasing the logic voltage and using smaller dies to accommodate thermal constraints would lower $ per op/s but make the server less energy efficient, with a high W per op/s. Die count per lane is also varied to find the optimal design.

Figure 4 shows an example of the design space exploration for a 28nm Bitcoin ASIC server. Each curve is associated with a particular die area and within each curve, the points from left to right represent voltage scaling from near-threshold to 50% above the nominal supply voltage, if junction temperatures are not exceeded. This figure is depicted for 9 ASICs per lane, which is the optimal for a 28nm Bitcoin server.

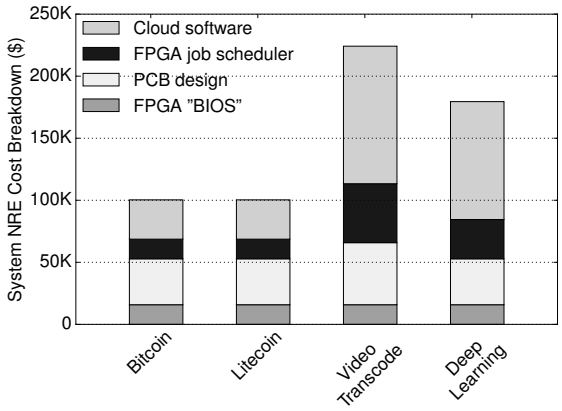


Figure 5: **System-level (non-ASIC) NRE varies based on PCB, Firmware and Cloud Software complexity.**

5.3 Benchmark Applications

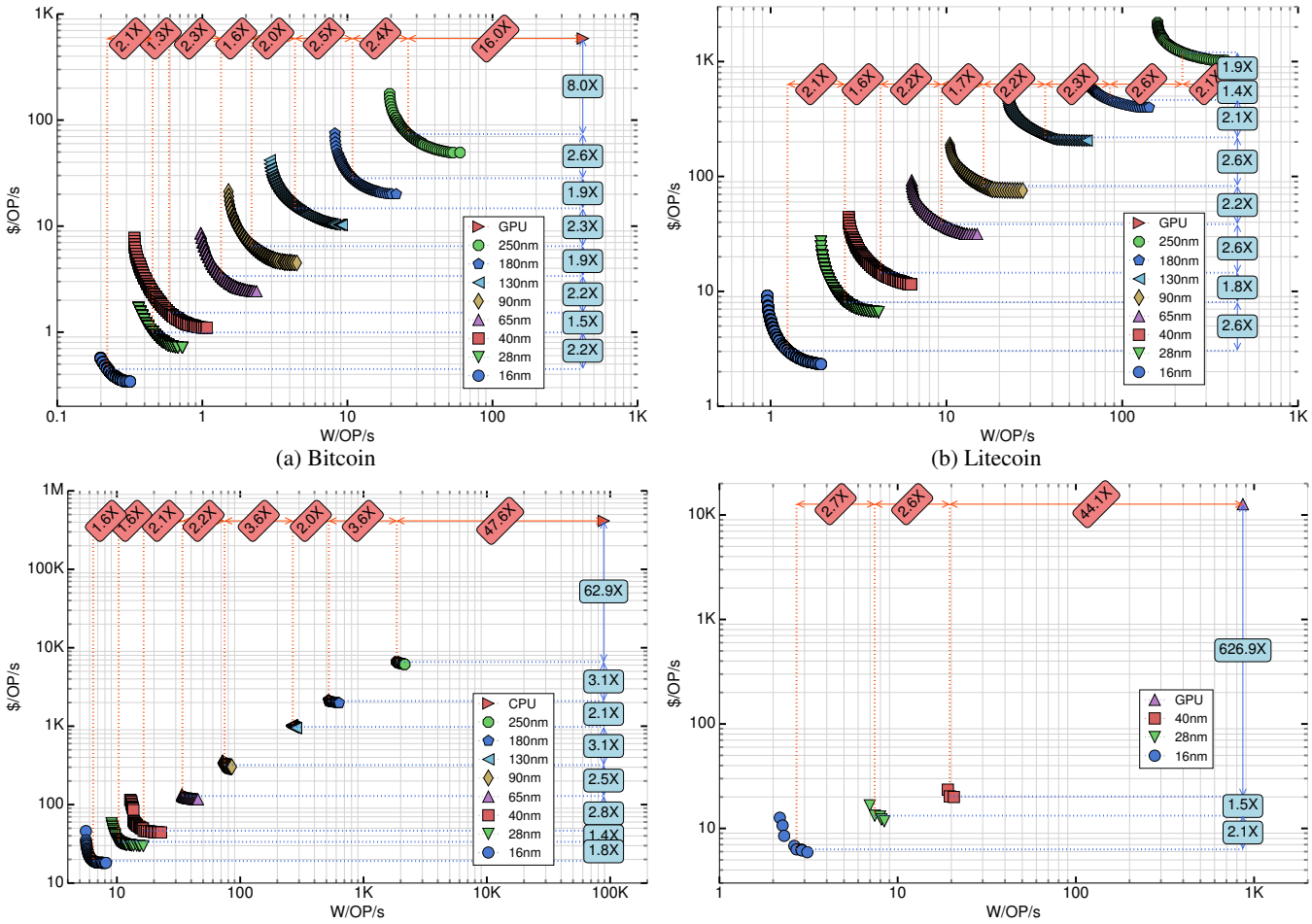
To explore ASIC Clouds across a range of accelerator properties, we examine four applications. Bitcoin is a very logic intensive design which has high power density and no need for SRAM or external DRAM. Litecoin is an SRAM-intensive application which has low power density. Video Transcoding represents an external memory-intensive application that needs DRAMs next to each ASIC, and also high off-PCB bandwidth. Finally, for an application that is latency sensitive we chose a Deep Learning neural net accelerator with a tight low-latency SLA. For Bitcoin and Litecoin, we developed the RCA and got the required parameters such as gate count from placed and routed designs in UMC 28nm using Synopsys IC compiler, and analysis tools (e.g. PrimeTime). We used H.265/HEVC Transcode design parameters from [31] and DaDianNao [13] (DDN) for the last two.

DDN proposed a scale-up system of 8x8 accelerator chips to be able to increase neural net size for Deep Learning application. We explored different server layouts in different technologies to be able to make such a 8x8 system, by considering 8 lanes per server and different die layouts that integrate multiple DDN chips as 1x1, 2x1, 2x2, 3x3 and 2x4 RCAs, as long as they fit in the mask reticle. This adds another design constraint and considering a DDN node as an RCA enables us to evaluate designs with high transistor count. Placing more DDN nodes per die saves in inter-die communication and the area and energy dedicated to DDN's HyperTransport I/O. We kept the frequency of these Deep Learning ASICs fixed across tech nodes to meet the SLA.

To model labor cost, shown in Table 5, we measured our development time for Bitcoin and Litecoin frontend to calculate the required man months (Mm). We estimated the DDN front-end labor based on the paper's description and our experience designing neural network accelerators. For Video Transcode, we assumed that the company already had an internally-developed encoder IP, but had to license the decoder for $200K. PCB design costs shown in Table 5 were

App	Cloud HW	Perf.	Power (W)	Cost ($)	TCO/ Op/s
Bitcoin	AMD 7970	0.68 GH/s	285	400	2,320
Bitcoin	28nm ASIC	8,223 GH/s	3,736	8.2K	2.9
Litecoin	AMD 7970 MH/s	0.63	285	400	2,500
Litecoin	28nm ASIC	1,384	3,662	11.2K	19.5
Video Transcode	Core-i7 4790K	0.0018 Kfps	155	725	791K
Video Transcode	28nm ASIC	158 Kfps	1,633	5.3K	78.5
Conv. Neural Net	NVIDIA Tesla K20X	0.26 TOps/s	225	3.3K	17,580
Conv. Neural Net	28nm ASIC	470 TOps/s	3,493	6.2K	44.3

Table 6: **ASIC Servers greatly outperform the best non-ASIC alternative in terms of TCO per op/s.**



(a) Bitcoin (b) Litecoin

(c) Video Transcode (d) Deep Learning

Figure 6: **Pareto frontiers improve in both energy and cost efficiency for newer technologies.** Each set of points represent the Pareto frontiers in each technology; dotted lines indicate the TCO-optimal Pareto point for each technology. Relative improvements in cost per performance and power per performance for the TCO-optimal points in each two consecutive technology nodes are indicated, and between the oldest evaluated node and baseline GPU/CPU.

based on vendor quotes, and the system-level coding costs came from Bitcoin mining software repository dates. FPGA firmware is estimated based on our implementation work with similar PCI-E and GigE bridges. Figure 5 shows the system-level, non-ASIC NRE costs for each application.

5.4 TCO per op/s improvement

Table 6 compares the ASIC Cloud results in 28nm fabrication technology for these four applications and the best non-ASIC server alternative. For Bitcoin, Litecoin, and Deep Learning there are GPU servers, but for Video Transcode the built-in Intel Xeon acceleration is used. Table 6 shows massive TCO per op/s improvement for ASIC Clouds to over-satisfy the two-for-two rule's second condition, but NRE, the first second remains to be addressed in the next two sections.

6. Computing Pareto-Optimal Designs for each Tech Node, and their NREs

Here we find the TCO per op/s optimal design for each technology node using the methodology in Section 5. We examine trends among different technologies for these Pareto-optimal designs. We also compute the NRE of each design in each node, so that in Section 7 we can determine which node is optimal considering total NRE+TCO.

6.1 Comparing Pareto-optimal points across nodes

Section 5.2 explored different designs to find the optimal TCO per op/s design. This evaluation is performed for different technology nodes and the Pareto-optimal points are selected according to $ per op/s and W per op/s, considering different die area values, operating voltages, and number of ASICs per lane. Figure 6 shows these points for all technologies and compares to baseline GPU/CPU server.

Among different technology nodes we have both energy per op and cost per performance benefits from going towards newer nodes. Due to smaller changes in nominal voltages among newer nodes than older ones, and also the dark silicon phenomenon [55, 21, 53, 52, 17, 50], the improvement among two consecutive technology nodes degrades for more

Tech	250nm	180nm	130nm	90nm	65nm	40nm	28nm	16nm
RCAs per Die	10	20	39	83	159	377	769	1,818
Die Area (mm^2)	559	579	588	600	599	540	540	420
Die Cost ($)	16	18	29	32	33	42	66	74
Dies/Server	120	120	120	120	120	120	72	48
Logic Vdd	1.081	0.857	0.654	0.563	0.517	0.433	0.459	0.424
Freq. (MHz)	37	54	77	93	100	121	149	169
GH/s	42	121	347	914	1,888	5,466	8,223	14,687
Power (W)	1,089	1,314	1,509	1,997	2,541	3,217	3,736	3,246
Cost (K$)	3.1	3.4	5.1	5.9	6.4	8.3	8.2	6.6
W/GH/s	26.21	10.83	4.350	2.183	1.346	0.589	0.454	0.221
$/GH/s	73.71	28.29	14.72	6.469	3.383	1.527	0.994	0.449
TCO/GH/s	186.2	74.55	33.68	15.88	9.115	4.039	2.912	1.378
NRE K$	561	602	790	1,054	1,194	1,845	2,760	6,451

Table 7: **Bitcoin TCO-optimal ASIC Server properties across tech nodes.** Bitcoin is extremely power dense resulting in TCO-optimal servers operating at very low voltages.

advanced nodes. Transistor density, operating frequency range and voltage range among different technologies impact the power density of the dies, while thermal extraction limit is technology-independent. Therefore the spread of Pareto-optimal points for each technology node is different. Also in most Pareto curves 28nm and 40nm are closer to each other compared to other adjunct technologies. They have the same nominal voltage which reduces the energy efficiency gains.

For Bitcoin, even 250nm shows a $\sim$12$\times$ TCO improvement over the baseline GPU. Litecoin is memory dominated and as a result the 45nm GPU surpasses 250nm by 1.9$\times$ in \$ per op/s, but underperforms by 2.1$\times$ in energy. In the end, the 250nm ASIC has superior TCO. Video Transcode benefits substantially from going to ASIC, since many Transcode units are placed per chip and accelerator performance and energy efficiency far outpaces CPUs, even at 250nm. Due to our assumption about SLA requirements for the Deep Learning application, older technology nodes than 40nm cannot be used. Therefore, the initial jump from GPU to ASIC is substantial, but comes at significant NRE.

6.2 Comparison of TCO-optimal designs across nodes

The TCO-optimal point in each curve is selected and detailed results for each of the applications in different technology nodes can be found in Tables 7, 8, 9, and 10. Looking across technology nodes in each application, we see a general trend of decreasing voltages. The explanation is as follows: the

Tech	40nm	28nm	16nm
RCA per Die	2x1	2x2	4x2
Die Area (mm^2)	259	298	195
Die Cost ($)	29	61	62
Dies/Server	32	64	80
Logic Vdd	1.285	0.900	0.615
Freq. (MHz)	607	606	617
TOps/s	118	470	1,176
Power (W)	2,312	3,493	3,184
Cost (K$)	2.4	6.2	7.4
W/TOps/s	19.60	7.431	2.708
$/TOps/s	20.25	13.25	6.304
TCO/TOps/s	100.4	44.28	17.78
NRE K$	3,259	4,301	8,616

Table 8: **Deep Learning TCO-optimal ASIC Server properties across tech nodes.** Frequency is kept constant for Deep Learning servers to satisfy SLA requirements.

Tech	250nm	180nm	130nm	90nm	65nm	40nm	28nm	16nm
RCAs per Die	12	22	47	98	188	446	910	2,150
Die Area (mm^2)	567	539	599	599	599	540	540	420
Die Cost ($)	16	17	29	32	32	42	66	74
Dies/Server	120	120	120	120	120	120	120	80
Logic Vdd	1.845	1.378	1.138	0.924	0.816	0.697	0.656	0.594
Freq. (MHz)	78	109	173	239	281	417	576	776
MH/s	2	6	21	62	139	491	1,384	2,938
Power (W)	516	525	769	1,000	1,298	2,068	3,662	3,664
Cost (K$)	2.8	2.9	4.6	5.1	5.3	7.1	11.2	9
W/MH/s	219.8	85.19	36.45	16.25	9.360	4.216	2.645	1.247
$/MH/s	1203	463.4	218.7	82.83	38.41	14.53	8.059	3.052
TCO/MH/s	2,214	854.8	388.5	156.8	79.97	32.94	19.49	8.353
NRE K$	591	633	835	1,104	1,254	1,924	2,823	6,404

Table 9: **Litecoin TCO-optimal ASIC Server properties across tech nodes.** Litecoin design is SRAM dominated with low power density, thus TCO optimal servers use a voltage close to technology nominal voltage to benefit from the available cooling opportunity to gain higher performance.

non-power limited performance of the silicon is improving because of transistor frequency increases and transistor count increases, scaling as S^3. At the same time, wafer costs (and thus cost per mm^2) are increasing by S, resulting in a net silicon potential improvement of S^2 in \$ per op/s. At the same time, capacitance is decreasing by S, improving energy efficiency by the same amount. TCO-optimal systems will balance improvements in \$ per op/s with improvements in W per op/s. Due to this S^2 versus S mismatch in CMOS scaling (especially with dark silicon, where nominal Vdd stops dropping), optimal designs drive the voltage further and further below nominal Vdd to make up for the difference.

How many ticks before a tock? Intel made famous the idea of separating process scaling (ticks) from architectural refactors (tocks). To understand the importance of data-center customization and co-sensitivity of architecture and process, we ported the TCO-optimal ASIC design in each process technology to future nodes, for example, the 250nm Bitcoin design was mapped from 180nm to 16nm. In this porting, RCAs count per ASIC is fixed. Only the operating voltage is changed to fit the thermal budget and then the TCO-optimal voltage is selected. DRAM count per ASIC for Video

Tech	250nm	180nm	130nm	90nm	65nm	40nm	28nm	16nm
RCAs per Die	2	5	9	19	37	92	153	140
DRAMs per Die	1	1	1	1	1	3	6	9
Die Area (mm^2)	493	634	627	619	623	594	498	177
Die Cost ($)	14	21	32	35	35	49	65	34
Dies/Server	64	64	64	64	64	64	40	32
Logic Vdd	2.533	1.818	1.501	1.171	1.015	0.957	0.754	0.710
Freq. (MHz)	56	77	115	165	215	358	429	705
Kfps	0.3	1.3	4	12	30	126	158	190
Power (W)	628	674	985	875	1,024	2,077	1,633	1,220
Cost (K$)	2.2	2.7	3.6	3.7	3.9	5.9	5.3	3.6
W/Kfps	1860	523.0	266.9	74.70	34.01	16.47	10.34	6.418
$/Kfps	6582	2094	978.8	319.6	128.7	46.50	33.56	19.13
TCO/Kfps	14,722	4411	2151	652.8	278.4	117.2	78.46	46.80
NRE K$	2,216	2,258	2,721	3,017	3,179	3,971	4,993	10,093

Table 10: **Video Transcode TCO-optimal ASIC Server properties across tech nodes.** Video Transcode optimal servers try to saturate DRAM bandwidth and trade-off operating voltage with RCAs per ASIC.

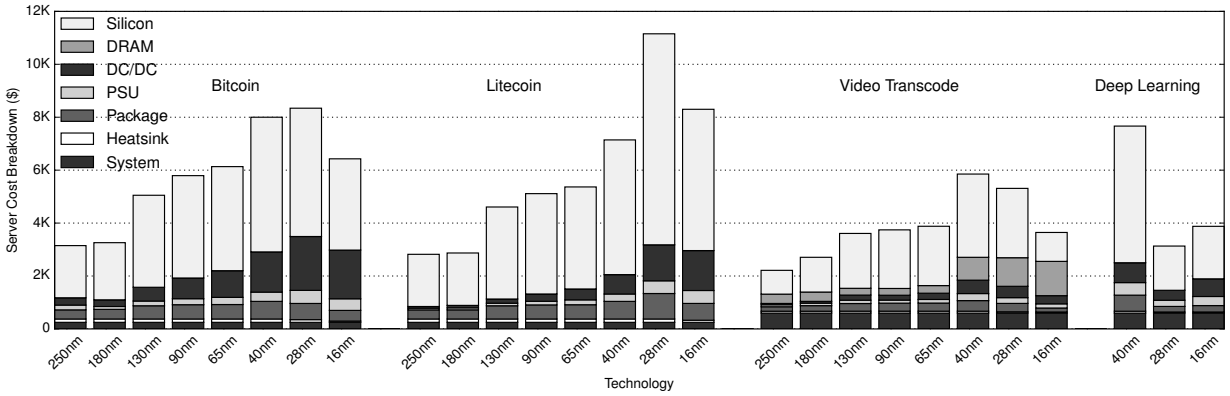


Figure 7: **Total server cost and relation of its components change across technology for TCO-optimal servers.** Server cost increases as more advanced technology nodes are used, except 16nm where power density becomes the limiting factor. Silicon is the dominant factor in server cost and newer nodes increase silicon utilization despite rising wafer costs. Package and power delivery components cost increase with tech node and remaining system costs stay relatively constant.

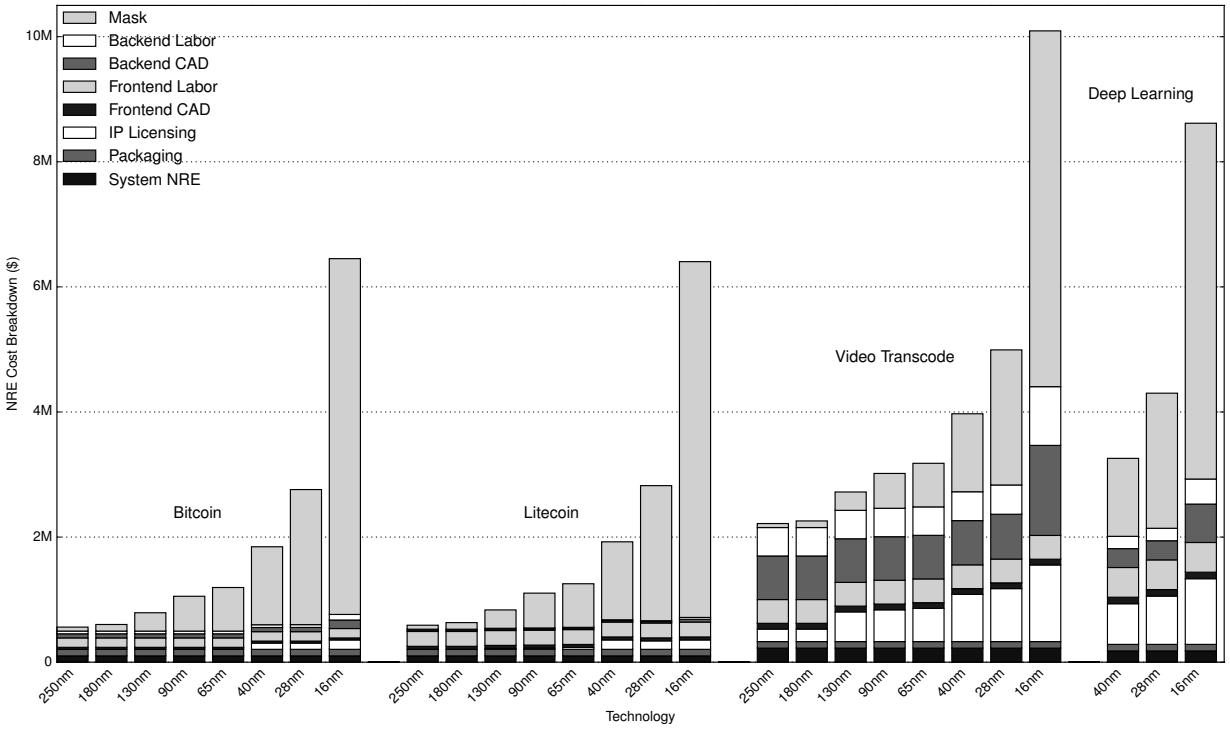


Figure 8: **NRE Cost Breakdown Across Tech Nodes.** Mask costs rise rapidly for newer technology nodes and become the dominant part of NRE. IP, CAD Tool and Labor Costs are application-dependent but can dominant mask costs in older nodes. Frontend labor and CAD is constant across nodes. IP costs for DDR and PCI-E/HyperTransport for newer nodes rises quickly.

Transcode and operating frequency for Deep Learning are also fixed, but the PCB is redesigned.

The farther the destination node is from the source node, the less optimal a ported design is compared to the TCO-optimal design for the destination node. Porting the optimal ASIC design in 250 nm to 16nm is worse in TCO by 2.14X, 3.68X, 6.71X for Litecoin, Bitcoin and Video Transcoding respectively. Porting the optimal ASIC design in 65nm to 16nm has worse TCO by 1.34X for Deep Learning.

On the other hand, porting across a single node leads to smaller TCO penalty: 1.05X in Bitcoin, 1.08X in Litecoin, and 1.06X in Deep Learning. For Video Transcode, designs are less stable, with a geo-mean of 1.53X at 65nm and above; and a geo-mean of 1.07X at 40nm and below. The large changes are due to successive DRAM technology improvements, ramping to LPDDR3 in 65nm.

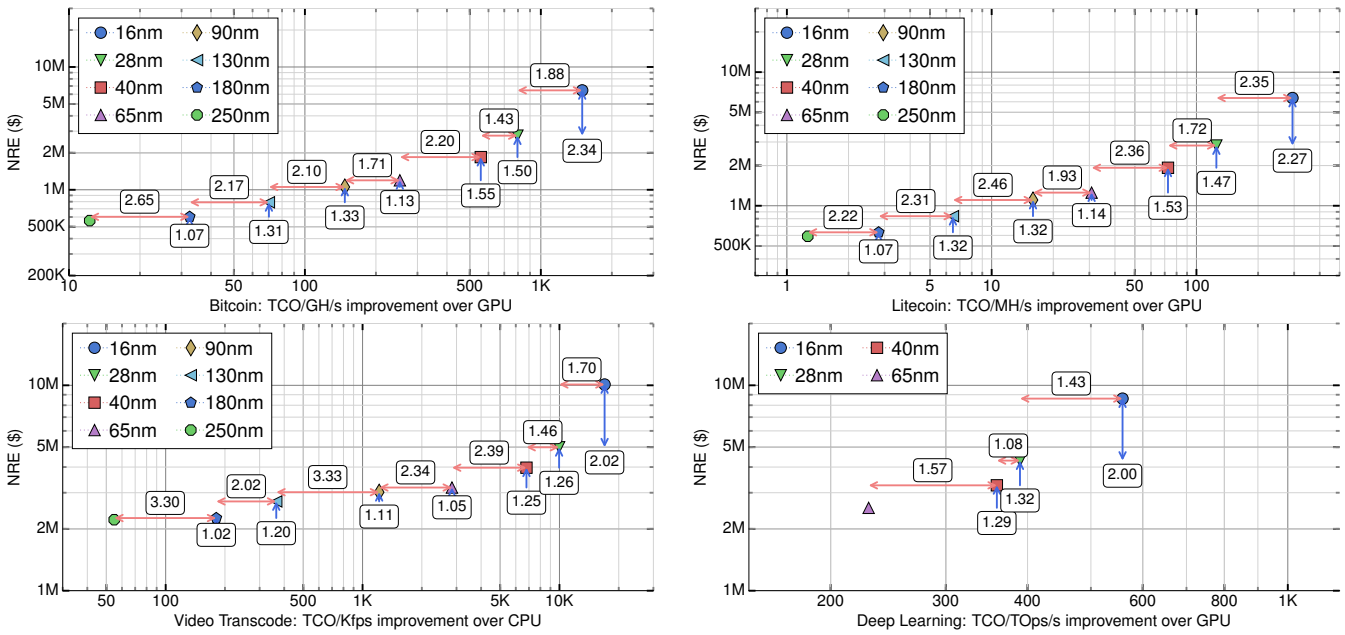


Figure 9: **Comparing marginal NRE and TCO per op/s improvements for each node.** Summary of NRE for different technology nodes versus TCO improvement over baseline.

6.3 Server Cost Change Across Tech Nodes

Server cost comprises die cost, power delivery components, cooling components, DRAMs and system components such as PCB. Figure 7 shows the server cost of the TCO-optimal servers in different technology nodes. System cost also includes fan costs, PCB cost, FPGA controller cost, and network card cost if required. These system costs stay relatively constant across technology nodes. Bitcoin and Litecoin server costs increase as tech nodes advance except for 16nm. Ultra-dense transistors in 16nm create thermal limits that reduce usable silicon per lane. However, TCO still improves.

Video Transcode server costs decline after 40nm. These servers are constrained by growing board space required by DRAMs required to feed each ASIC. 40nm and older have 8 ASICs per lane and 1-3 DRAMs, but TCO-optimal 28nm ASICs saturate 6 DRAMs each and only 5 ASICs fit in a lane. 16 nm ASICs rise to 9 DRAMs and only 4 ASICs fit. Still these configurations are more energy and cost efficient, despite the growing ratio of DRAM to silicon cost. In 28nm and 16nm there are 240 and 288 DRAMs per server respectively, compared to 192 DRAMs in 40nm. Designs in 130nm, 90nm and 65nm cannot saturate a single DRAM's bandwidth and the DRAM cost remains constant. Finally for 250nm and 180nm, DRAM cost increases marginally due to use of SDRAM instead of LPDDR, derived by lack of DDR IP availability and NRE cost savings.

Deep Learning has large RCAs and very limited layout options which worsen the power density issue, especially in 28nm. This eliminated some TCO-optimal points due to small violations of thermal hotspots. To address this problem, we added dark silicon to spread hotspots. For example, the TCO-optimal design in 28nm uses $40mm^2$ or 15.5% extra silicon per die to be able to have more ASICs per lane. This marginal increase in silicon cost pays off because fixed parts of system cost are amortized over more RCAs per server. Since the ASIC's operating voltage in 16nm is lowered to match the SLA requirement, power density issues are mitigated and no dark silicon is necessary. This enables more silicon per lane and makes the 16nm servers cost more than 28nm ones.

6.4 NRE Calculation across Tech Nodes

Based on the NRE model described in Section 4, the NRE cost breakdown across nodes and applications is shown in Figure 8. The trend clearly shows that the overall NRE cost rapidly increases as technology node advances and that mask costs for newer nodes become the dominant part of NRE.

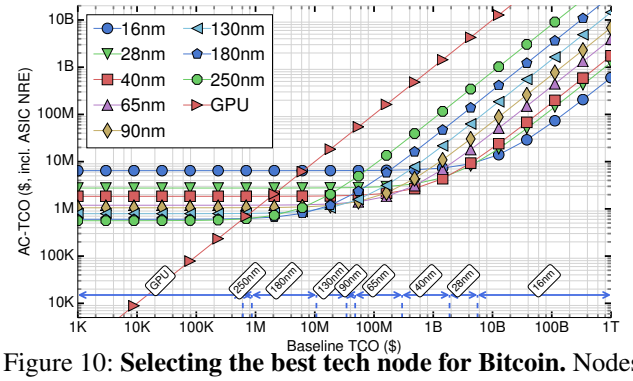


Figure 10: **Selecting the best tech node for Bitcoin.** Nodes start at higher total cost but eventually become the lowest TCO for a period, resulting in savings. White boxes represent the best node for a range of TCO for Bitcoin.

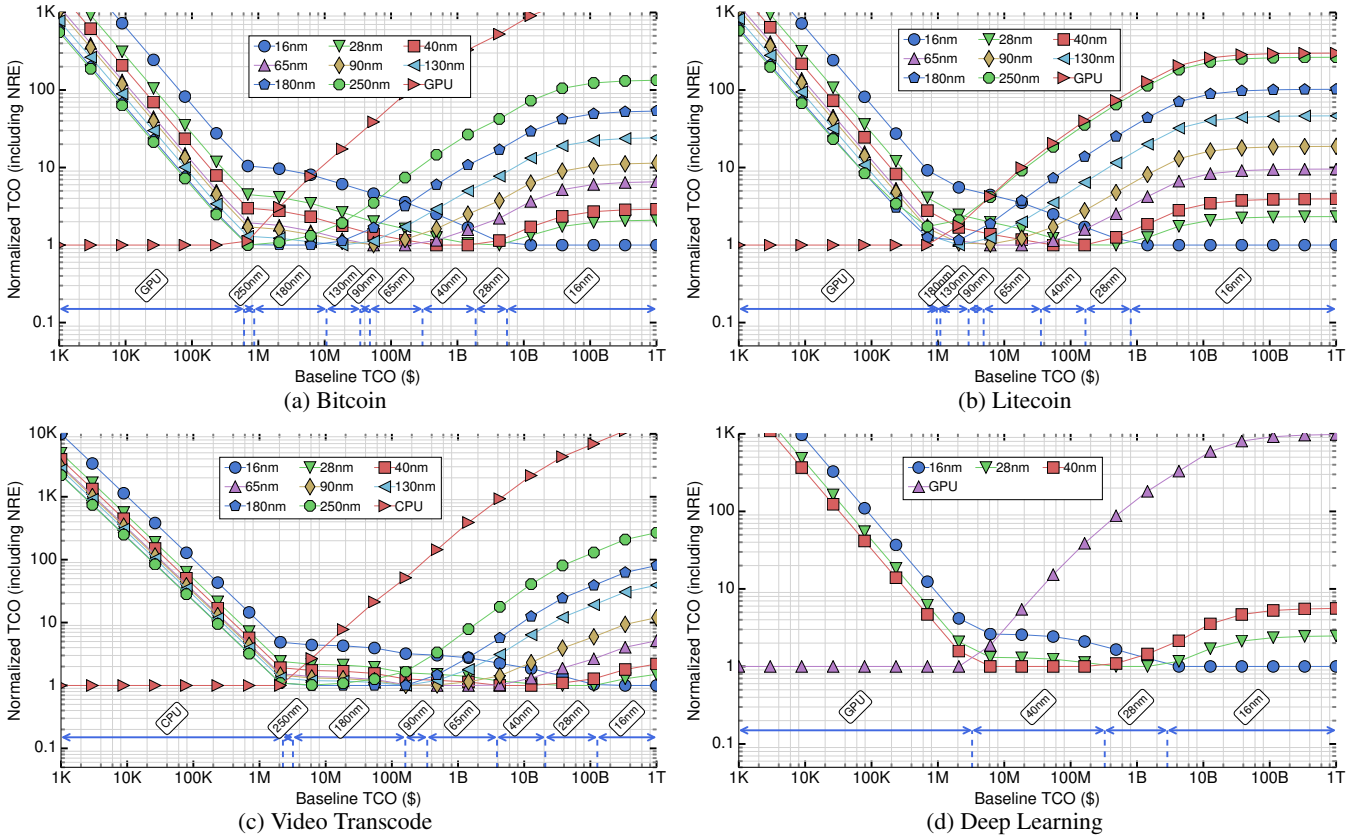


Figure 11: **For these applications, cutting edge node technologies become TCO-optimal only for extreme-scale ASIC Clouds.** For each total cost value, the best server scenario is set to 1 and values for other scenarios represent the ratio of total money spent. After the break-even point with baseline system, there is an enormous money saving opportunity.

Labor, tool costs, IP costs and system NRE vary widely between applications but with the exception of backend labor in 16nm and PHY IP, is relatively constant across nodes. Figures 3 and 8 show how IP prices scale across tech nodes.

7. Finally, NRE+TCO Optimal ASIC Clouds

In this section, we finally combine our NRE and TCO analyses from the previous section to derive NRE+TCO optimal ASIC Clouds.

7.1 TCO per op/s improvement versus NRE increases

To understand how NRE and TCO per op/s scale together across nodes, Figure 9 shows relative NRE and TCO per op/s changes among different technology nodes. From 250nm to 65nm, TCO per op/s increases more rapidly than NRE costs. After 65nm, the slope fundamentally changes, as NRE increases more rapidly, and TCO per op/s performance improvements flatten as described in Section 6.1.

7.2 Optimal nodes given pre-/post- ASIC Cloud TCO

As Cloud application demand increases, the baseline TCO spent on non-ASIC servers increase and creates the opportunity of going towards ASIC servers, based on the Two-for-Two rule. Figure 10 presents total cost for a variety of ASIC Cloud implementations of Bitcoin across tech nodes. The arrows indicate ranges of optimality for an ASIC Cloud implemented in a given node. For example, when TCO of using GPU servers exceeds $610K, 250nm becomes the least expensive option. Similarly, when the TCO reaches $867K, then 180nm becomes the least expensive option, and so on to $1.9B for 28nm and $5.6B for 16nm.

To compare nodes, Figure 11 shows the normalized version of Figure 10 where for each pre-ASIC Baseline TCO, we divide by TCO of the best technology node. The arrows in these graphs indicate, given an input TCO, what node should be targeted to minimize TCO including NRE.

For example, for Bitcoin, Litecoin, and Video Transcode, 180nm is optimal for small TCO's from $860K-$10.6M, $960K-$1.1M, and $3.2M-$160M respectively. Deep Learning's SLA requires >= 40nm, which is optimal from $3M to $326M. 130nm and 90nm have narrow applicability.

7.3 Picking the node

A company can simply plug in their forecasts for the demand and baseline TCO of the application to determine what node to use to minimize NRE+TCO. Accelerator researchers have less clarity on what TCO to use for their proposed ASIC Cloud. They could estimate the application's demand in a

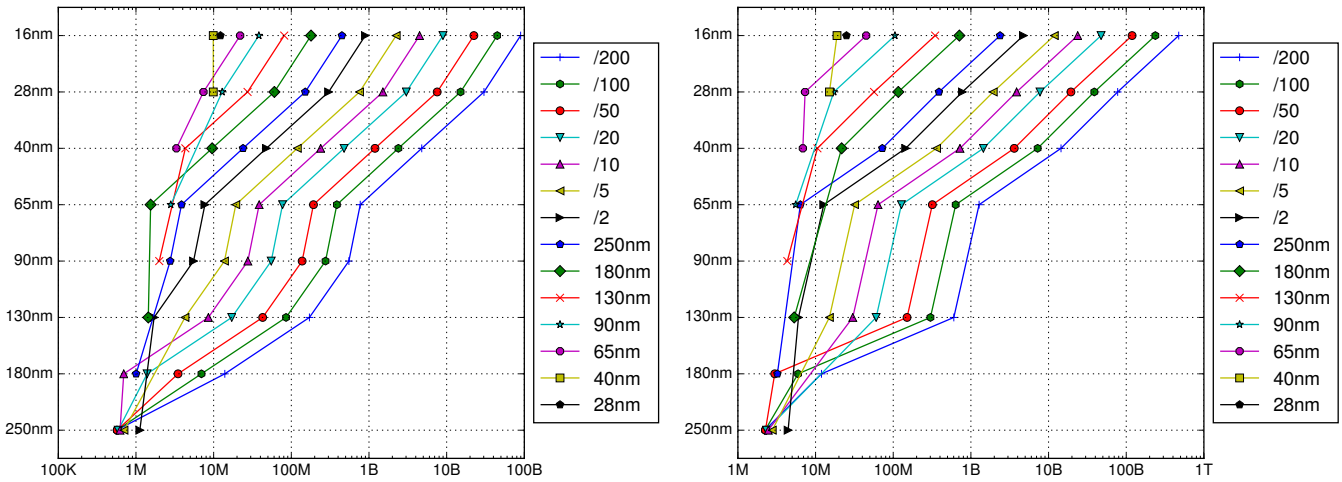


Figure 12: **Optimal node selection based on TCO (X axis) and on *tech parity node* (shown in key): the tech node closest to the pre-accelerated version's TCO per op/s.** Applications with small IP NRE (like Bitcoin) are represented on left and medium amounts of IP (like Video Transcode) on right. Parity nodes of */N* indicate $N\times$ the TCO per op/s of 250nm.

datacenter, just like the company, which would allow them to select a node. However, in some cases, these parameters are difficult to estimate because the application is emerging and there is not yet demand.

7.4 Advanced nodes like 16 nm not always better

Our data suggests that using the latest node (e.g. 16nm) for an emerging datacenter accelerator can be a mistake. For example, our results show that 16nm was optimal only for TCOs starting at $805M for Litecoin and reaching a geomean of $6.36B across all four applications. Effectively, by choosing an advanced node to do a study, a researcher is setting too high of an NRE on the technology, preventing a prospective company or investor from adopting the technology. Rather, the optimal node must provide *just enough* TCO improvement over the baseline. Moreover, reduced NREs allow an ASIC Cloud to be more agile, updating ASICs more frequently to track evolving software.

7.5 Tech Parity Nodes: Getting to *Just Enough*

To address this issue, we introduce the *tech parity node*, the technology node at which an ASIC would have similar TCO per op/s to the best alternative. Using this formalism, and knowing the estimated TCO of the workload, Figure 12 gives the target node that best reduces TCO+NRE. For example, for a low-IP-NRE Bitcoin-like app, if the parity node is 250nm (key), and the emerging computation has a $25M TCO (x-axis), then 40nm would be a reasonable target node (y-axis).

8. Related Work

Recent architectural work on accelerators. Over the years, there have been many studies that examine the design of systems based on accelerators or application specific chips. Some of the oldest work is the GF11 [9] scale-up physics simulator. Anton [48] targeted scale-up molecular dynamics simulation. More recent work that targets accelerators for a broad spectrum of application domains, including neural networks [27, 43, 14, 35, 38, 4, 46, 16, 64, 29, 10], big data [32, 23], 3D Ultrasound [45], graph analytics [40, 25, 3], databases [36, 60], key-value stores [24, 37], natural processing language [51], regular expressions [20], speech recognition [28, 62], irregular integer applications like SpecInt [55, 56, 44], Android Hotspots [21, 22], gzip compression [2], H.264 encode [26] and convolution [42].

Warehouse-scale scale-out acceleration. [54] examined the ramp from FPGA to GPU to ASIC Cloudhardware for Bitcoin. Microsoft Catapult [41, 12] proposed FPGA-based clouds. Magaki et al [39] proposed ASIC Clouds. Baymax [15] examined non-premptive accelerators in Clouds.

Design NRE reduction. Recent efforts have focused on reducing cost by creating chip generators, languages and estimators [49, 6, 47] and leveraging pre-built systems for ASIC bringup [57].

IP and mask NRE reduction. Recent open source HW efforts [5, 7, 18] propose a path to reduced IP costs. Mask NRE reduction techniques have been proposed at different levels covering manufacturing and assembling. Kim et al. proposed to build SoCs out of pre-existing libraries of custom chiplets [34]. Structured ASICs try to reduce NRE [58, 59, 63], but with significant penalties.

9. Conclusion

We hope that this paper will lead to reduced TCO and greater varieties of ASIC Clouds in the future. The paper's models can be found at `darksilicon.net/nre`.

Acknowledgments

This work was partially supported by NSF Award 1228992 and AMD Gifts, and by STARnet's Center for Future Architectures Research, a SRC program sponsored by MARCO and DARPA. We thank Partha Ranganathan and the reviewers for their helpful comments. We thank eSilicon Corporation for their support and Geoff Porter in particular for guidance in the NRE model.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: a system for large-scale machine learning. In *OSDI*, 2016.

[2] M. Abdelfattah, A. Hagiescu, and D. Singh. Gzip on a chip: High performance lossless data compression on FPGAs using opencl. In *International Workshop on OpenCL (IWOC*, 2014.

[3] J. Ahn, S. Hong, S. Yoo, O. Mutlu, and K. Choi. A Scalable Processing-in-memory Accelerator for Parallel Graph Processing. In *ISCA*, 2015.

[4] J. Albericio, P. Judd, T. Hetherington, T. Aamodt, N. Jerger, and A. Moshovos. Cnvlutin: ineffectual-neuron-free deep neural network computing. In *ISCA*, 2016.

[5] K. Asanovic, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz, S. Karandikar, B. Keller, D. Kim, and J. Koenig. The Rocket Chip Generator. Technical Report No. UCB/EECS-2016-17, 2016.

[6] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Avizienis, J. Wawrzynek, and K. Asanovic. Chisel: Constructing hardware in a Scala embedded language. In *DAC*, 2012.

[7] J. Balkind, M. McKeown, Y. Fu, T. Nguyen, Y. Zhou, A. Lavrov, M. Shahrad, A. Fuchs, S. Payne, X. Liang, M. Matl, and D. Wentzlaff. OpenPiton: An Open Source Manycore Research Framework. In *ASPLOS*, 2016.

[8] L. Barroso, J. Clidaras, and U. Holzle. *The Datacenter As a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second Edition*. Synthesis Lectures on Computer Architecture, 2013.

[9] J. Beetem, M. Denneau, and D. Weingarten. The GF11 Supercomputer. In *ISCA*, 1985.

[10] M. Bojnordi, and E. Ipek. Memristive Boltzmann Machine: A Hardware Accelerator for Combinatorial Optimization and Deep Learning. In *HPCA*, 2016.

[11] I. Bolsens. 2.5 D ICs: Just a Stepping Stone or a Long Term Alternative to 3D?. Keynote Talk at 3-D Architectures for Semiconductor Integration and Packaging Conference, 2011.

[12] A. Caulfield, E. Chung, A. Putnam, H. Angepat, J. Fowers, M. Haselman, S. Heil, M. Humphrey, P. Kaur, J. Kim, D. Lo, T. Massengill, K. Ovtcharov, M. Papamichael, L. Woods, S. Lanka, D. Chiou, and D. Burger A Cloud-Scale Acceleration Architecture. In *MICRO*, 2016.

[13] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, and O. Temam. DaDianNao: A Machine-Learning Supercomputer. In *MICRO*, 2014.

[14] Y. Chen, J. Emer, and V. Sze. Eyeriss: a spatial architecture for energy-efficient dataflow for convolutional neural networks. In *ISCA*, 2016.

[15] Q. Chen, H. Yang, J. Mars, and L. Tang. Baymax: QoS Awareness and Increased Utilization for Non-Preemptive Accelerators in Warehouse Scale Computers. In *ASPLOS*, 2016.

[16] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie. PRIME: a novel processing-in-memory architecture for neural network computation in ReRAM-based main memory. In *ISCA*, 2016.

[17] H. Esmaeilzadeh, E. Blem, R. Amant, K. Sankaralingam, and D. Burger. Dark Silicon and the End of Multicore Scaling. In *ISCA*, 2011.

[18] V. Gangadhar, R. Balasubramanian, M. Drumond, Z. Guo, J. Menon, C. Joseph, R. Prakash, S. Prasad, P. Vallathol, and K. Sankaralingam. MIAOW: An open source GPGPU. In *IEEE Hot Chips 27 Symposium*, 2015.

[19] Glassdoor. Glassdoor salaries, 2016. `https://www.glassdoor.com`

[20] V. Gogte, A. Kolli, M. Cafarella, L. D'Antoni, and T. Wenisch. HARE: Hardware accelerator for regular expressions. In *MICRO*, 2016.

[21] N. Goulding, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, J. Babb, M. Taylor, and S. Swanson. GreenDroid: A mobile application processor for a future of dark silicon. In *IEEE Hot Chips 22 Symposium*, 2010.

[22] N. Goulding-Hotta, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, P. Huang, M. Arora, S. Nath, V. Bhatt, J. Babb, S. Swanson, and M. Taylor. The GreenDroid Mobile Application Processor: An Architecture for Silicon's Dark Future. In *IEEE MICRO*, 2011.

[23] B. Gu, A. Yoon, D. Bae, I. Jo, J. Lee, J. Yoon, J. Kang, M. Kwon, C. Yoon, S. Cho, J. Jeong, and D. Chang. Biscuit: a framework for near-data processing of big data workloads. In *ISCA*, 2016.

[24] A. Gutierrez, M. Cieslak, B. Giridhar, R. G. Dreslinski, L. Ceze, and T. Mudge. Integrated 3D-stacked Server Designs for Increasing Physical Density of Key-value Stores. In *ASPLOS*, 2014.

[25] T. Ham, L. Wu, N. Sundaram, N. Satish, and M. Martonosi. Graphicionado: A High-Performance and Energy-Efficient Accelerator for Graph Analytics. In *MICRO*, 2016.

[26] R. Hameed, W. Qadeer, M. Wachs, O. Azizi, A. Solomatnikov, B. Lee, S. Richardson, C. Kozyrakis, and M. Horowitz. Understanding sources of inefficiency in general-purpose chips. In *ISCA*, 2012.

[27] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. Horowitz, and W. Dally. EIE: Efficient Inference Engine on Compressed Deep Neural Network. In *ISCA*, 2016.

[28] J. Hauswald, M. Laurenzano, Y. Zhang, C. Li, A. Rovinski, A. Khurana, R. Dreslinski, T. Mudge, V. Petrucci, L. Tang, and J. Mars. Sirius: An Open End-to-End Voice and Vision Personal Assistant and Its Implications for Future Warehouse Scale Computers. In *ASPLOS*, 2015.

[29] Y. Ji, Y. Zhang, S. Li, P. Chi, C. Jiang, P. Qu, Y. Xie, and W. Chen NEUTRAMS: Neural Network Transformation and Co-design under Neuromorphic Hardware Constraints. In *MICRO*, 2016.

[30] H. Jones. Strategies in Optimizing Market Positions for Semiconductor Vendors Based on IP Leverage. IBS White Paper, 2014.

[31] C. Ju, T. Liu, K. Lee, Y. Chang, H. Chou, C. Wang, T. Wu, H. Lin, Y. Huang, C. Cheng, T. Lin, C. Chen, Y. Lin, M. Chiu, W. Li, S. Wang, Y. Lai, P. Chao, C. Chien, M. Hu, P. Wang, Y. Huang, S. Chuang, L. Chen, H. Lin, M. Wu, and C. Chen. A 0.5 nJ/Pixel 4 K H.265/HEVC Codec LSI for Multi-Format Smartphone Applications. In *JSSC*, 2016.

[32] S. Jun, M. Liu, S. Lee, Hicks, Ankcorn, King, Myron, S. Xu, and Arvind. BlueDBM: An Appliance for Big Data Analytics. In *ISCA*, 2015.

[33] A. Kannan, N. Jerger, and G. Loh. Enabling Interposer-based Disintegration of Multi-core Processors. In *MICRO*, 2015.

[34] M. Kim, M. Mehrara, M. Oskin, and T. Austin. Architectural Implications of Brick and Mortar Silicon Manufacturing. In *ISCA*, 2007.

[35] D. Kim, J. Kung, S. Chai, S. Yalamanchili, and S. Mukhopadhyay. Neurocube: A Programmable Digital Neuromorphic Architecture with High-Density 3D Memory. In *ISCA*, 2016.

[36] O. Kocberber, B. Grot, J. Picorel, B. Falsafi, K. Lim, and P. Ranganathan. Meet the Walkers: Accelerating Index Traversals for In-memory Databases. In *MICRO*, 2013.

[37] K. Lim, D. Meisner, A. Saidi, P. Ranganathan, and T. Wenisch. Thin Servers with Smart Pipes: Designing SoC Accelerators for Memcached. In *ISCA*, 2013.

[38] S. Liu, Z. Du, J. Tao, D. Han, T. Luo, Y. Xie, Y. Chen, and T. Chen. Cambricon: An Instruction Set Architecture for Neural Networks. In *ISCA*, 2016.

[39] I. Magaki, M. Khazraee, L. Vega, M. B. Taylor. ASIC Clouds: Specializing the Datacenter. In *ISCA*, 2016.

[40] M. Ozdal, S. Yesil, T. Kim, A. Ayupov, J. Greth, S. Burns, and O. Ozturk. Energy efficient architecture for graph analytics accelerators. In *ISCA*, 2016.

[41] A. Putnam, A. Caulfield, E. Chung, D. Chiou, K. Constantinides, J. Demme, H. Esmaeilzadeh, J. Fowers, G. Gopal, J. Gray, M. Haselman, S. Hauck, S. Heil, A. Hormati, J. Kim, S. Lanka, J. Larus, E. Peterson, S. Pope, A. Smith, J. Thong, P. Xiao, and D. Burger. A Reconfigurable Fabric for Accelerating Large-scale Datacenter Services. In *ISCA*, 2014.

[42] W. Qadeer, R. Hameed, O. Shacham, P. Venkatesan, C. Kozyrakis, and M. Horowitz. Convolution engine: balancing efficiency and flexibility in specialized computing. In *ISCA*, 2013.

[43] B. Reagen, P. Whatmough, R. Adolf, S. Rama, H. Lee, S. Lee, J. Hernández-Lobato, G. Wei, and D. Brooks. Minerva: enabling low-power, highly-accurate deep neural network accelerators. In *ISCA*, 2016.

[44] J. Sampson, G. Venkatesh, N. Goulding-Hotta, S. Garcia, S. Swanson and M. Taylor. Efficient Complex Operators for Irregular Codes. In *HPCA*, 2011.

[45] R. Sampson, M. Yang, S. Wei, C. Chakrabarti, and T. Wenisch. Sonic Millip3De: A Massively Parallel 3D-Stacked Accelerator for 3D Ultrasound. In *HPCA*, 2013.

[46] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. Strachan, M. Hu, R. Williams, and V. Srikumar. ISAAC: a convolutional neural network accelerator with in-situ analog arithmetic in crossbars. In *ISCA*, 2016.

[47] Y. Shao, B. Reagen, G. Wei, and D. Brooks. Aladdin: a Pre-RTL, power-performance accelerator simulator enabling large design space exploration of customized architectures. In *ISCA*, 2014.

[48] D. Shaw, M. Deneroff, R. Dror, J. Kuskin, R. Larson, J. Salmon, C. Young, B. Batson, K. Bowers, J. Chao, M. Eastwood, J. Gagliardo, J. Grossman, C. Ho, D. Ierardi, I. Kolossváry, J. Klepeis, T. Layman, C. McLeavey, M. Moraes, R. Mueller, E. Priest, Y. Shan, J. Spengler, M. Theobald, B. Towles, and S. Wang. Anton, a Special-purpose Machine for Molecular Dynamics Simulation. In *ISCA*, 2007.

[49] A. Solomatnikov, A. Firoozshahian, W. Qadeer, O. Shacham, K. Kelley, Z. Asgar, M. Wachs, R. Hameed, and M. Horowitz. Chip Multi-processor Generator. In *DAC*, 2007.

[50] A. Pedram, S. Richardson, S. Galal, S. Kvatinsky, and M. Horowitz. Dark Memory and Accelerator-Rich System Optimization in the Dark Silicon Era. In *IEEE Design Test*, 2016.

[51] P. Tandon, J. Chang, R. Dreslinski, V. Qazvinian, P. Ranganathan, and T. Wenisch. Hardware Acceleration for Similarity Measurement in Natural Language Processing. In *ISLPED*, 2013.

[52] M. Taylor. A Landscape of the New Dark Silicon Design Regime. In *IEEE Micro*, 2013.

[53] M. Taylor. Is Dark Silicon Useful? Harnessing the Four Horsemen of the Coming Dark Silicon Apocalypse. In *DAC*, 2012.

[54] M. Taylor. Bitcoin and the Age of Bespoke Silicon. In *CASES*, 2013.

[55] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. Taylor. Conservation cores: reducing the energy of mature computations In *ASPLOS*, 2010.

[56] G. Venkatesh, J. Sampson, N. Goulding-Hotta, S. Kota Venkata, M. Taylor, and S. Swanson. QsCores: Configurable Co-processors to Trade Dark Silicon for Energy Efficiency in a Scalable Manner. In *MICRO*, 2011.

[57] M. Wachs, O. Shacham, Z. Asgar, A. Firoozshahian, S. Richardson and M. Horowitz. Bringing up a chip on the cheap. *IEEE Design Test of Computers*, 2012.

[58] J. Wong, F. Kourshanfar and M. Potkonjak. Flexible ASIC: shared masking for multiple media processors. In *DAC*, 2005.

[59] K. Wu, and Y. Tsai. Structured ASIC, Evolution or Revolution?. In *Proceedings of the International Symposium on Physical Design (ISPD)*, 2004.

[60] L. Wu, A. Lottarini, T. Paine, M. Kim, and K. Ross. Q100: The Architecture and Design of a Database Processing Unit. In *ASPLOS*, 2014.

[61] N. Xu, X. Cai, R. Gao, L. Zhang, and F. Hsu. FPGA Acceleration of RankBoost in Web Search Engines. In *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2009.

[62] R. Yazdani, A. Segura, J. Arnau, and A. Gonzalez. An ultra low-power hardware accelerator for automatic speech recognition. In *MICRO*, 2016.

[63] B. Zahiri. Structured ASICs: opportunities and challenges. In *Proceedings of the 21st International Conference on Computer Design (ICCD)*, 2003.

[64] S. Zhang, Z. Du, L. Zhang, H. Lan, S. Liu, L. Li, Q. Guo, T. Chen, and Y. Chen. Cambricon-X: An accelerator for sparse neural networks. In *MICRO*, 2016.