Government, University, & Industry Cooperation
The NVIDIA Story

National Science Board

July 23, 2025

Bill Dally

Chief Scientist and SVP of Research, NVIDIA Corporation

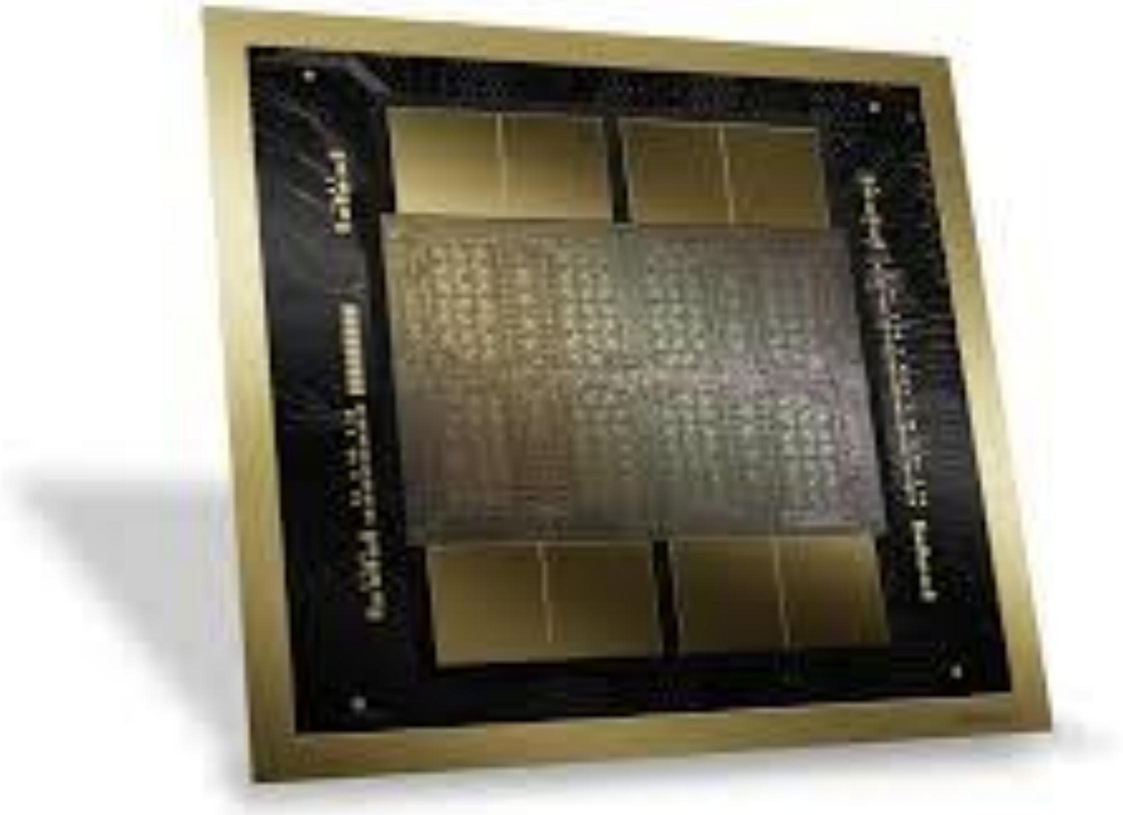
Adjunct Professor of CS and EE, Stanford

Government Funding → University Research → Trained People / Technology

Great Companies

NVIDIA

Technology Leadership

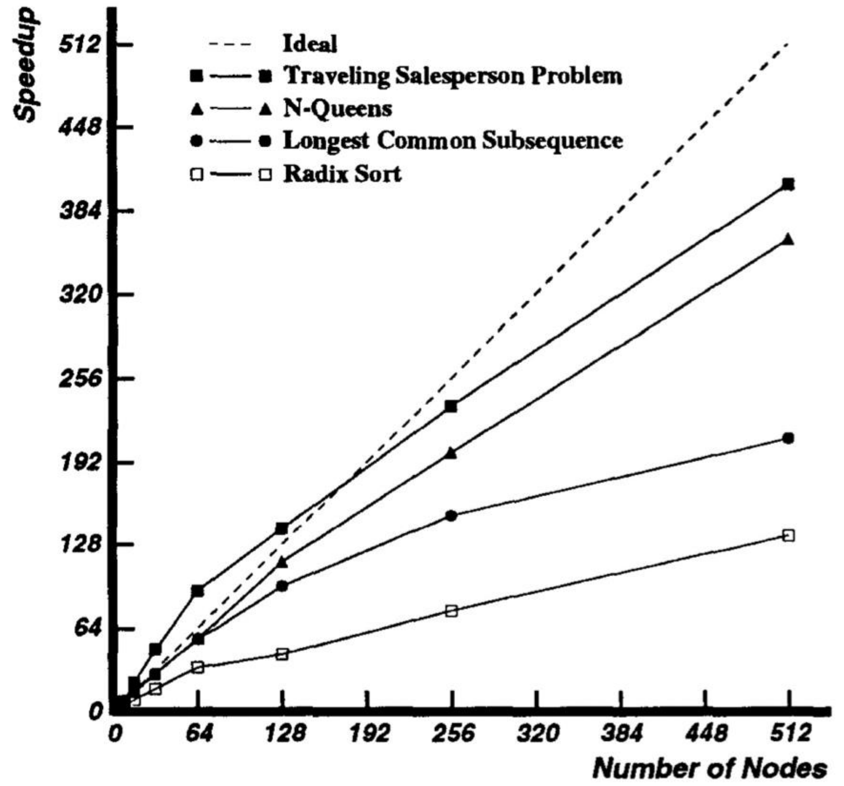



2001 Streaming Supercomputer
Brook, GPU Computing
DOE Funded
2003 Consulting on NV50
2004 Ian Buck to NVIDIA
2006 G80 & CUDA
Not to scale
2017 Volta
NVSwitch
TensorCores
Summit Supercomputer
1997 Stream Processing
Kernels, P&C Locality
DARPA Funded
2012 AlexNet
1992 MIT J-Machine
Synch & Comm Mechanisms
DARPA Funded
2012 Titan Supercomputer
2010 Andrew Ng Project
Leads to CuDNN
1983 Work on Caltech Cosmic Cube
Fundamentals of Parallel Computing
DARPA Funded
2009 Bill Joins NVIDIA Full Time

The J-Machine: A Retrospective

William J. Dally[1], Andrew Chang[1], Andrew Chien[2], Stuart Fiske[3], Waldemar Horwat[4], John Keen[3], Richard Lethin[5], Michael Noakes, Peter Nuth[6], Ellen Spertus[7], Deborah Wallach[8], D. Scott Wills[9]

Eleven years ago, at ISCA 14, we published a paper titled, "Architecture of a Message-Driven Processor" [Dal87] marking the start of our J-Machine project at MIT. The project culminated with the construction of a working



...ne evaluation of this ... ne use of a *jellybean* ... a processor, memory, ... as a building block for ... rain parallel computer ... s of parallelism by ... etween processor and ... small set of efficient ... *mechanisms* that were ... ogramming models. It ... ging without software ... e dispatch a message

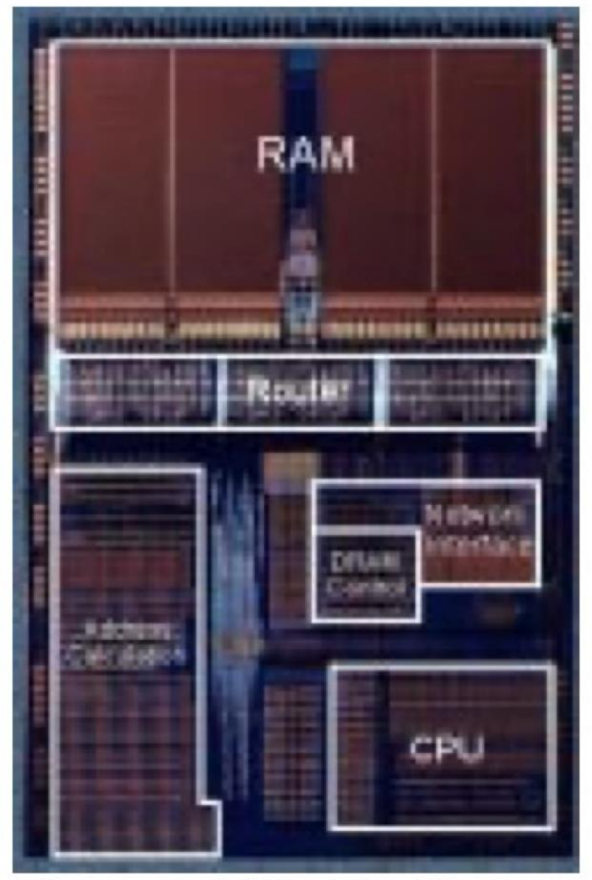


Figure 1: MDP Die Photo

Figure 2: J-Machine

revising the chips in early 1992 to correct a few bugs, we built three J-Machines: a 1024-node machine at MIT and

Dally, William J., et al. "Retrospective: the J-machine." *25 years of the international symposia on Computer architecture (selected papers)*. 1998.

ICCD2002

The Imagine Stream Processor

Ujval J. Kapasi, William J. Dally, Scott Rixner†, John D. Owens, and Brucek Khailany *

Computer Systems Laboratory
Stanford University, Stanford, CA 94305, USA
{ujk,billd,jowens,khailany}@cva.stanford.edu

†Computer Systems Laboratory
Rice University, Houston, TX 77005, USA
rixner@rice.edu

Abstract

The Imagine Stream Processor is a single-chip programmable media processor with 48 parallel ALUs. At 400 MHz, this translates to a peak arithmetic rate of 16 GFLOPS on single-precision data and 32 GOPS on 16-bit fixed-point data. The scalability of Imagine's programming model and architecture enable it to achieve such high arithmetic rates. Imagine executes applications that have been mapped to the stream programming model. *The stream model decomposes applications into a set of computation kernels that operate on data streams. This mapping exposes the inherent locality and parallelism in the application and*

on a chip. This is because both providing instructions and transferring data at the necessary rates are problematic. For example, a 48 ALU single-chip processor must issue up to 48 instructions/cycle and provide up to 144 words/cycle of data bandwidth to operate at peak rate.

The *Imagine Stream Processor* addresses these issues by using the stream programming model to expose parallelism as well as producer-consumer locality, the true data locality in media processing applications. This locality can be exploited by routing most of the required bandwidth on local wires, which are more efficient and plentiful than global communication paths. Imagine exploits this locality with a

*The research described in this paper was supported by the Defense Advanced Research Projects Agency under ARPA order E254 and monitored by the Army Intelligence Center under contract DABT63-96-C0037, by ARPA order L172 monitored by the Department of the Air Force under contract F29601-00-2-0085, by Intel Corporation, by Texas Instruments, by an Intel Foundation Fellowship, and by the Interconnect Focus Center Program for Gigascale Integration under DARPA Grant MDA972-99-1-0002.

ACM Queue 2004

Stream Processors

Programmability with Efficiency

WILLIAM J. DALLY, UJVAL J. KAPASI, BRUCEK KHAILANY, JUNG HO AHN, AND ABHISHEK DAS, STANFORD UNIVERSITY

Brook for GPUs: Stream Computing on Graphics Hardware

Ian Buck Tim Foley Daniel Horn Jeremy Sugerman Kayvon Fatahalian Mike Houston Pat Hanrahan
Stanford University

Abstract

In this paper, we present Brook for GPUs, a system for general-purpose computation on programmable graphics hardware. Brook extends C to include simple data-parallel constructs, enabling the use of the GPU as a streaming coprocessor. We present a compiler and runtime system that abstracts and virtualizes many aspects of graphics hardware. In addition, we present an analysis of the effectiveness of the GPU as a compute engine compared to the CPU, to determine when the GPU can outperform the CPU for a particular algorithm. We evaluate our system with five applications, the SAXPY and SGEMV BLAS operators, image segmentation, FFT, and ray tracing. For these applications, we demonstrate that our Brook implementations perform comparably to hand-written GPU code and up to seven times faster than their CPU counterparts.

CR Categories: I.3.1 [Computer Graphics]: Hardware Architecture—Graphics processors D.3.2 [Programming Languages]: Language Classifications—Parallel Languages

Keywords: Programmable Graphics Hardware, Data Parallel Computing, Stream Computing, GPU Computing, Brook

1 Introduction

In recent years, commodity graphics hardware has rapidly evolved from being a fixed-function pipeline into having programmable vertex and fragment processors. While this new modern hardware. In addition, the user is forced to express their algorithm in terms of graphics primitives, such as textures and triangles. As a result, general-purpose GPU computing is limited to only the most advanced graphics developers.

This paper presents *Brook*, a programming environment that provides developers with a view of the GPU as a streaming coprocessor. The main contributions of this paper are:

- The presentation of the Brook stream programming model for general-purpose GPU computing. Through the use of streams, kernels and reduction operators, Brook abstracts the GPU as a streaming processor.
- The demonstration of how various GPU hardware limitations can be virtualized or extended using our compiler and runtime system; specifically, the GPU memory system, the number of supported shader outputs, and support for user-defined data structures.
- The presentation of a cost model for comparing GPU vs. CPU performance tradeoffs to better understand under what circumstances the GPU outperforms the CPU.

2 Background

2.1 Evolution of Streaming Hardware

Programmable graphics hardware dates back to the original programmable framebuffer architectures [England 1986]. One of the most influential programmable graphics systems

TOG 2004

Ian Buck c. 2003

2004 – Brook language ported to GPUs

2004 - Ian Buck graduates and joins NVIDIA works with John Nickolls on CUDA

Brook + Cg + user commments -> CUDA

2006 – CUDA Launched

THEME ARTICLE: MICROPROCESSOR AT 50

Evolution of the Graphics Processing Unit (GPU)

William J. Dally and Stephen W. Keckler, *NVIDIA Corporation, Santa Clara, CA, 95051, USA*
David B. Kirk, *Independent Consultant*

Graphics processing units (GPUs) power today's fastest supercomputers, are the dominant platform for deep learning, and provide the intelligence for devices ranging from self-driving cars to robots and smart cameras. They also generate compelling photorealistic images at real-time frame rates. GPUs have evolved by adding features to support new use cases. NVIDIA's GeForce 256, the first GPU, was a dedicated processor for real-time graphics, an application that demands large amounts of floating-point arithmetic for vertex and fragment shading computations and high memory bandwidth. As real-time graphics advanced, GPUs became programmable. The combination of programmability and floating-point performance made GPUs attractive for running scientific applications. Scientists found ways to use early programmable GPUs

Computer Corporation was founded in 1968 to build special-purpose 3-D graphics hardware. Using the small-scale integration technology of the day, these expensive multitrack systems were used for demanding applications such as flight simulators.

THE AVAILABILITY OF EASILY PROGRAMMED GPUs WITH HIGH FLOATING-POINT PERFORMANCE ENABLED THE CURRENT REVOLUTION IN DEEP LEARNING.

IEEE Micro Nov-Dec 2021

Government Funding → University Research

University Research → Trained People

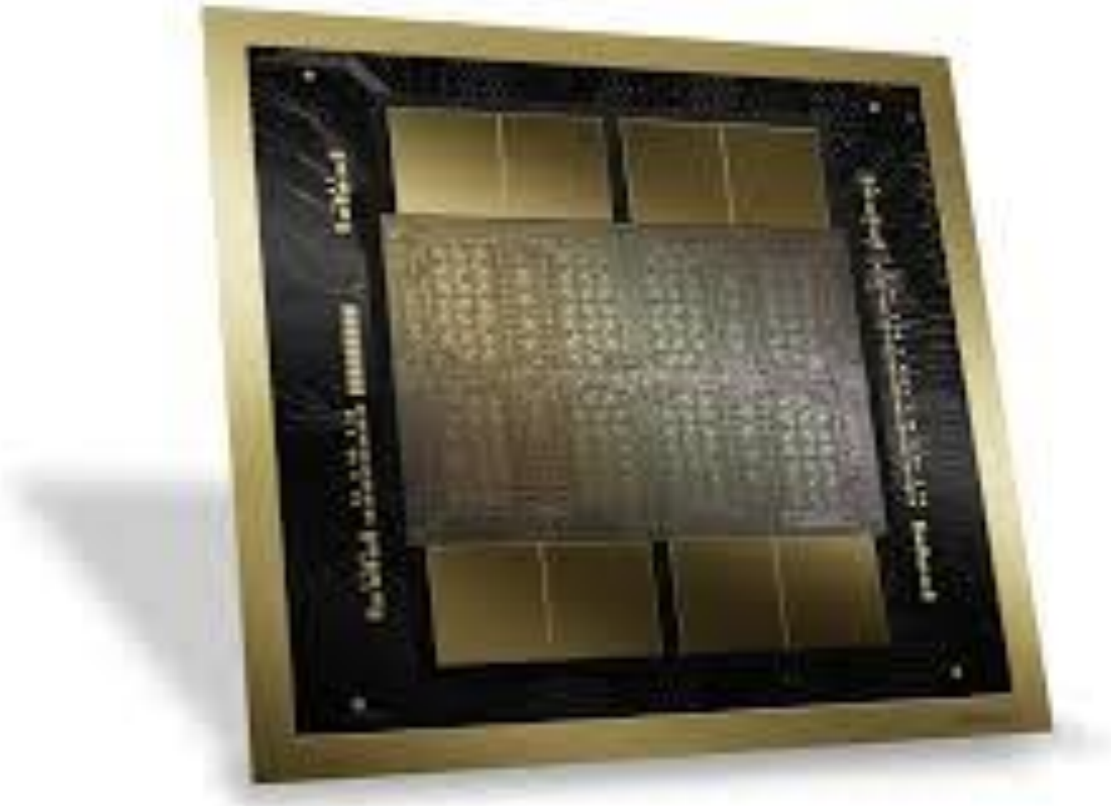
University Research → Technology

Trained People, Technology →

Great Companies

NVIDIA

Technology Leadership



Federally-funded university research plants the seeds for industrial success and U.S. leadership

Basic research solves problems beyond the horizon of industrial research labs

University research trains students in key technology areas

Foreign graduate students are attracted to universities working on leading research and often stay in the US

We are in an international competition – for technology and talent. We are ahead, but near-peer nations are closing the gap