Amber SoC Tapeout Overview

Stanford University -- AHA Monthly Meeting

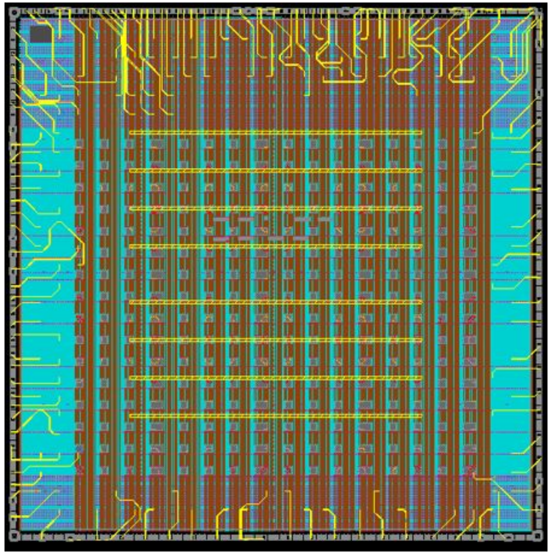
December 3, 2020

Christopher Torng, Taeyoung Kong

AHA
Agile Hardware Center

AHA SoC Evolution -- Jade CGRA

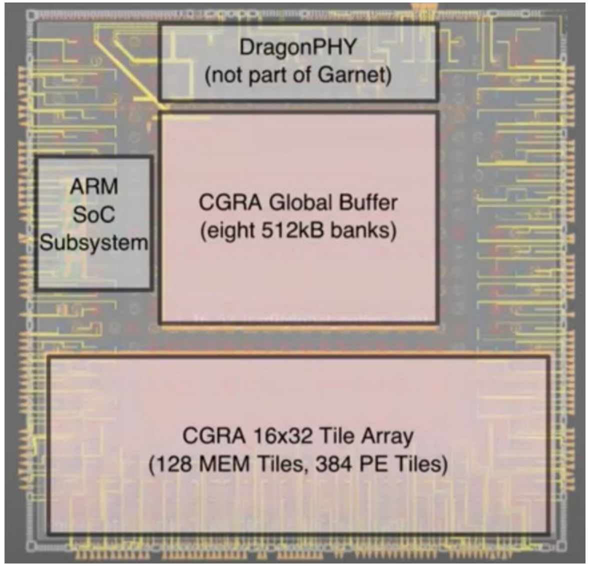
- 16x16 array of PE and memory tiles
- Simple 16b fixed point PE tiles
- Memory tiles with line buffer access pattern only
- Image processing applications



AHA
Agile Hardware Center

AHA SoC Evolution -- Garnet SoC

- Virtually taped out on June 2020
- Jade PE + **FP multiply and add (BFloat16)**
- Memory tiles + **general affine access pattern**
- Image processing + ML

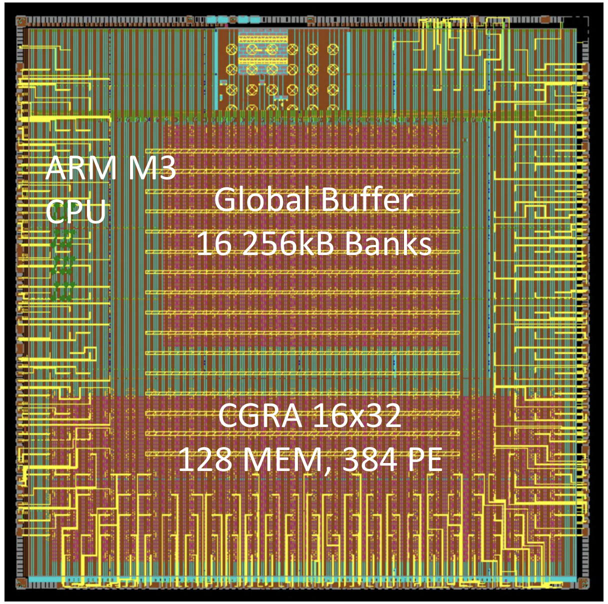


AHA
Agile Hardware Center

AHA SoC Evolution -- Amber SoC

- Taped out on December 1, 2020
- Garnet PE + **Local register file (ponds)**
- Garnet memory tiles + **Easier to program from the compiler**
- Verified fast reconfiguration, application partitioning, temporal and spatial multiplexing of applications
- Image processing and ML

TSMC 16nm
Cadence VDE
5mm x 5mm



AHA Agile Hardware Center

Amber SoC Architecture

Memory System for Virtualization and Energy-Efficiency

AHA
Agile Hardware Center

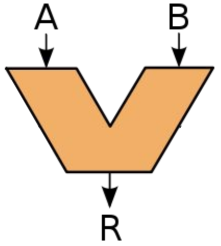
Tape-out Revisions and Memory System

Tapeout Revision	Hierarchy	Features
Jade (Gen-1 CGRA)	Dram -> Memory Tile	First version. No SoC
Garnet (Gen-2 CGRA + SoC)	Dram -> Global Buffer -> Memory Tile	Energy-efficiency with 3-level memory system
Amber SoC (Gen-3 CGRA + SoC)	Dram -> Global Buffer -> Memory Tile -> Pond (Compute-local RF)	**Virtualization** Energy-efficiency with 4-level memory system

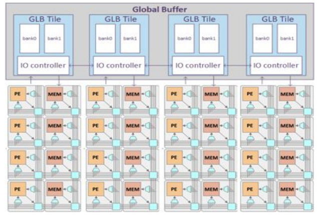
AHA
Agile Hardware Center

Introduction to CGRA Virtualization

- CGRA virtualization at different levels



Block-level



CGRA-level

Multi node level

- Definition / Advantages
  - Abstract CGRA hardware resources
  - Simplify the interface
  - Support flexible execution models

=> Requires hardware and software support

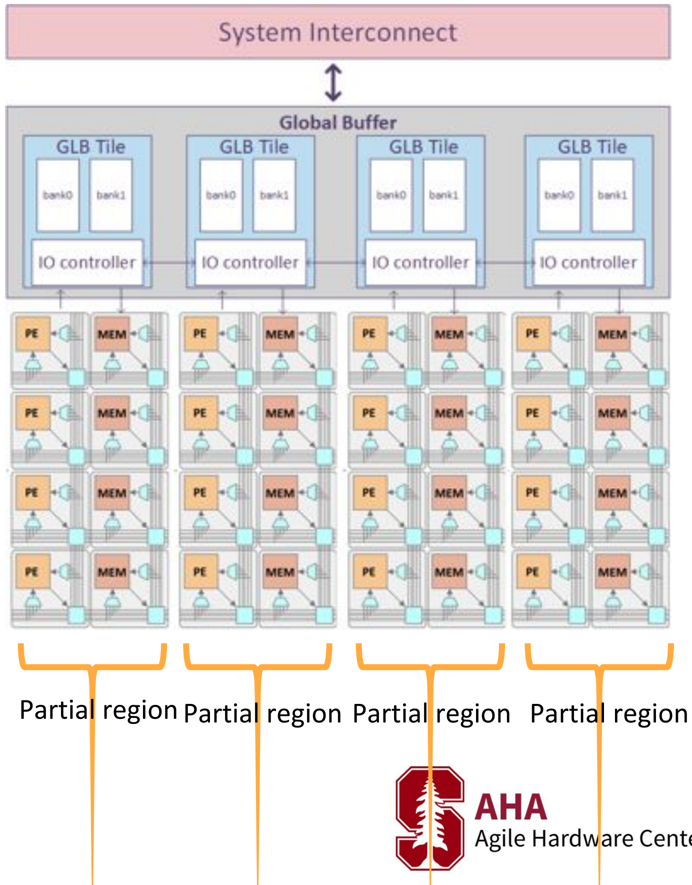
AHA
Agile Hardware Center

Amber: An Initial Prototype with Hardware Mechanisms for Virtualization

- Mechanism #1 : Run multiple kernels in parallel

- Mechanism #2: Relocatable bitstream to different partial regions

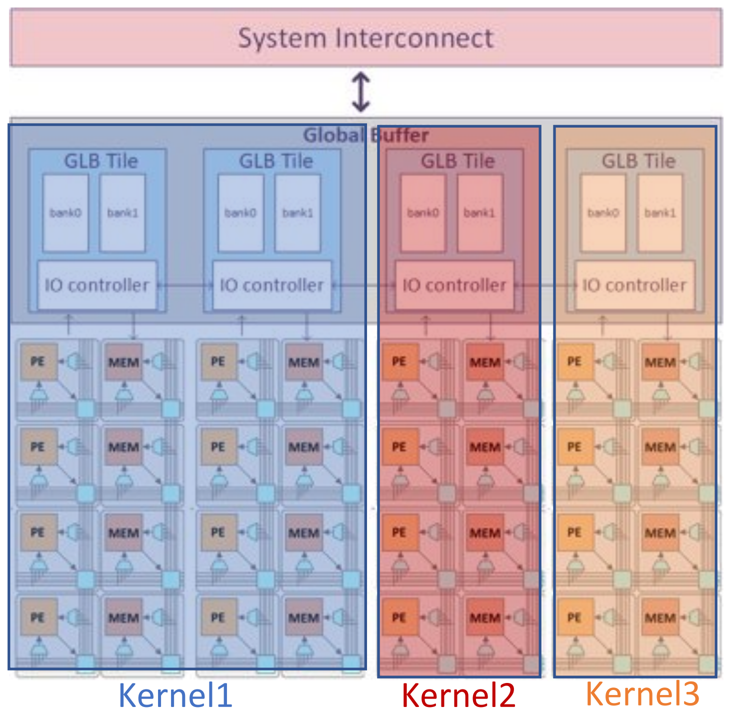
- Mechanism #3: Dynamic partial reconfiguration



AHA
Agile Hardware Center

Hardware Mechanism #1 – Multiple Kernels in Parallel

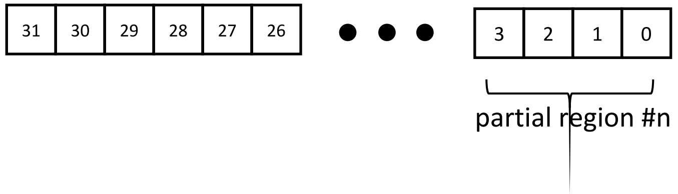
- Three kernels running independently and simultaneously
  - Virtualization at I/O, GLB, and Tile
- I/O allocation
  - Not limited to the location of the memory bank
- memory/computation allocation
  - MEM: Different # of global buffer
  - COMP: Different # of partial regions



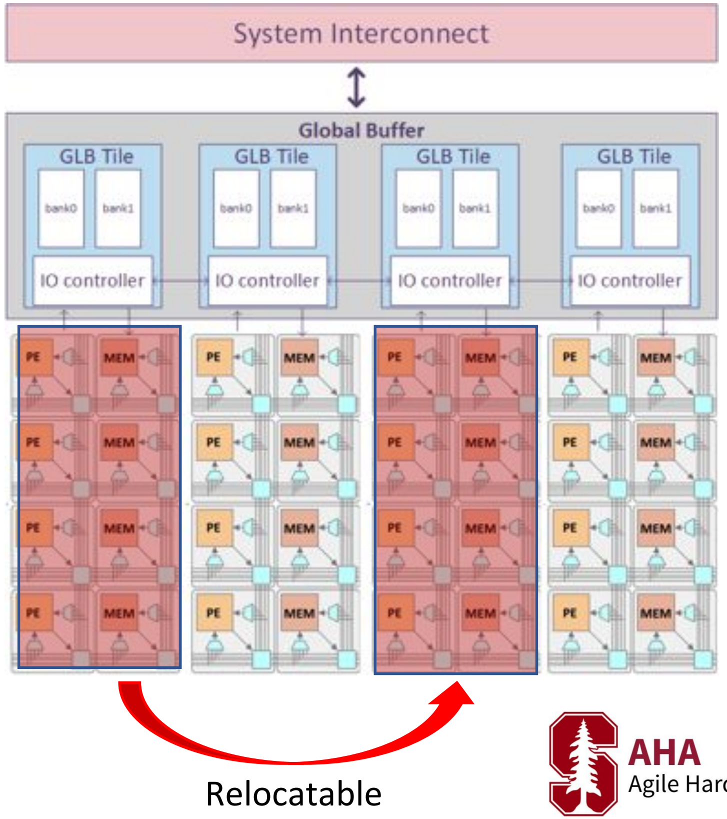
AHA Agile Hardware Center

Hardware Mechanism #2 – Relocatable Bitstream to Different Partial Regions

- Bitstream address breakdown



  - LSB 8bits indicate the column #
  - Rest address bits and data bits remain unchanged

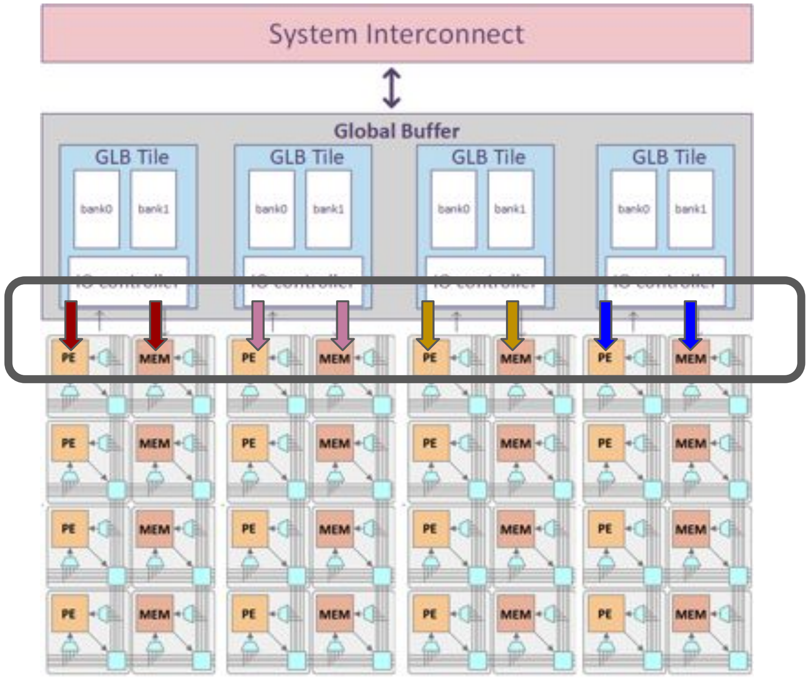


AHA
Agile Hardware Center

Hardware Mechanism #3 – Dynamic Partial Reconfiguration (DPR)

- Use global buffer to double buffer bitstream
  - Fast
  - Parallel
  - Independent

- Configuration time (75,520 registers)
  - Jtag: 108.7ms
  - AXI4-Lite: 604us
  - DPR: 4.72us



AHA
Agile Hardware Center

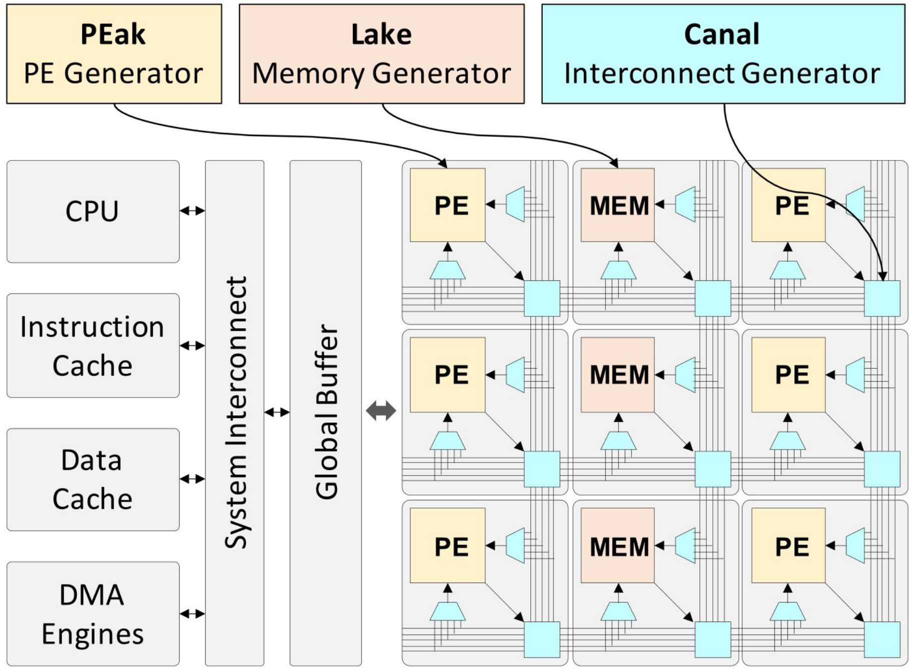
Tape-out Revisions and Memory System

Tapeout Revision	**Hierarchy**	**Features**
Jade (Gen-1 CGRA)	Dram -> Memory Tile	First version. No SoC
Garnet (Gen-2 CGRA + SoC)	Dram -> Global Buffer -> Memory Tile	Energy-efficiency with 3-level memory system
Amber SoC (Gen-3 CGRA + SoC)	Dram -> Global Buffer -> Memory Tile -> Pond (Compute-local RF)	Virtualization **Energy-efficiency with 4-level memory system**

AHA
Agile Hardware Center

Hardware Support for Energy-Efficient Memory System

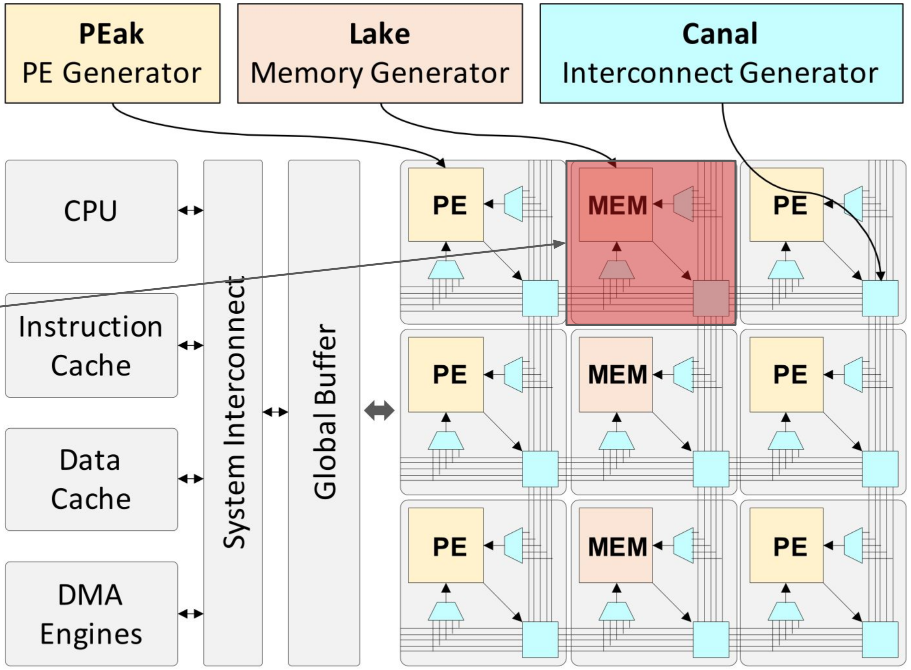
- More energy-efficient memory system
  - Reduce memory access
  - Memory hierarchy
  - Unified framework for each level of hierarchy



Hardware Support for Energy-Efficient Memory System

- More energy-efficient memory system
  - Reduce memory access
  - Memory hierarchy
  - Unified framework for each level of hierarchy

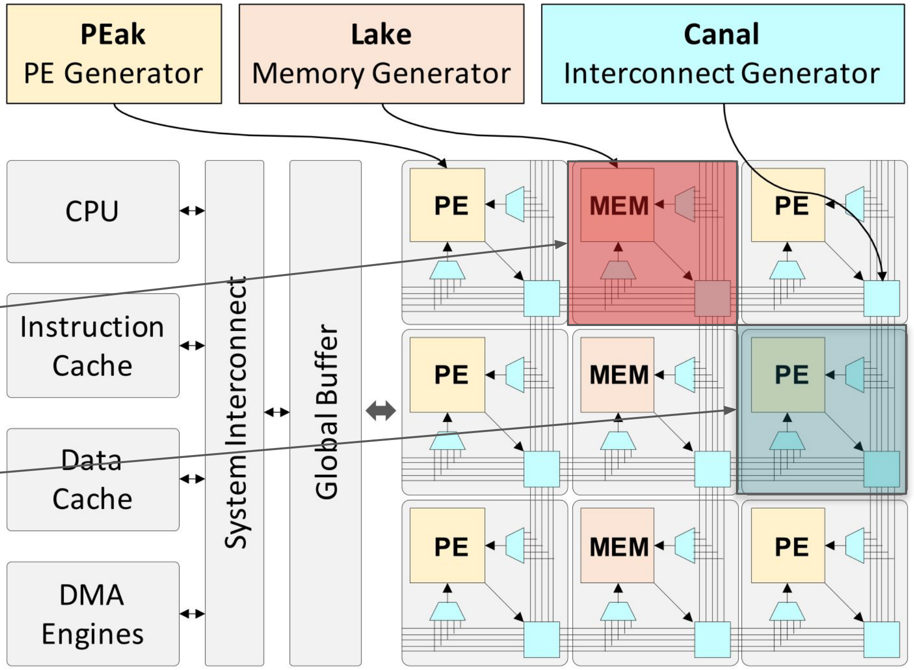
- Mechanism 1: Leverage wide-fetch, single-port SRAM for better efficiency



AHA
Agile Hardware Center

Hardware Support for Energy-Efficient Memory System

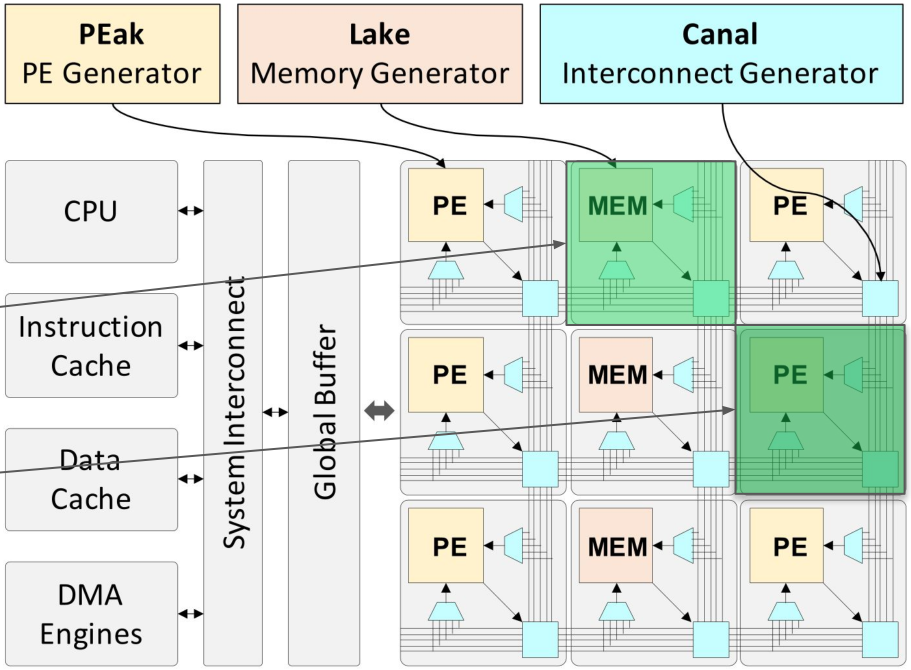
- More energy-efficient memory system
  - Reduce memory access
  - Memory hierarchy
  - Unified framework for each level of hierarchy
- Mechanism 1: Leverage wide-fetch, single-port SRAM for better efficiency
- Mechanism 2: Place small register-based memories in the PE



AHA
Agile Hardware Center

Hardware Support for Energy-Efficient Memory System

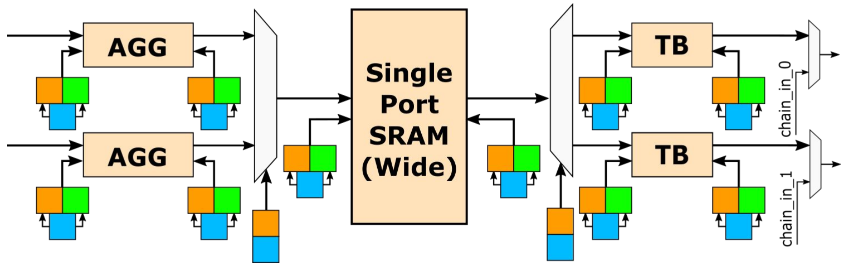
- More energy-efficient memory system
  - Reduce memory access
  - Memory hierarchy
  - Unified framework for each level of hierarchy
- Mechanism 1: Leverage wide-fetch, single-port SRAM for better efficiency
- Mechanism 2: Place small register-based memories in the PE
- Mechanism 3: Unified memory generator framework for every level of hierarchy



AHA Agile Hardware Center

Hardware Mechanism #1 - Wide Fetch SRAM

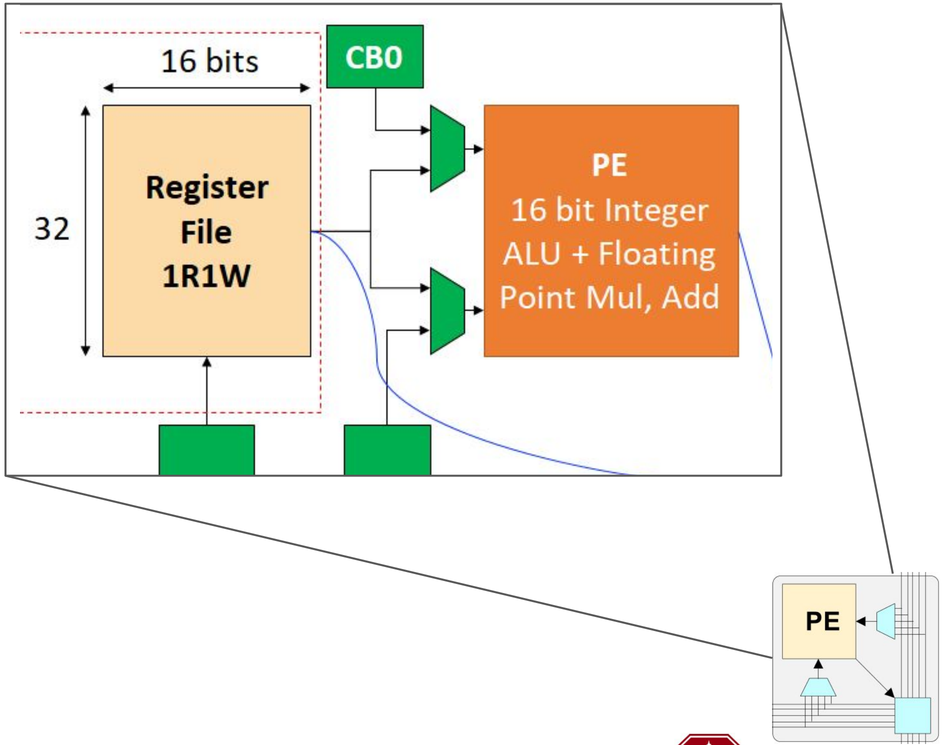
- Mechanism 1: Leverage wide-fetch, single-port SRAM
  - Reduce memory read/write
- Wide-fetch SRAMs exhibit better energy per data accessed compared to single-wide SRAMs
- Wide-fetch SRAMs provide equivalent bandwidth with only a modest control overhead compared to Dual-Ported SRAMs.



AHA Agile Hardware Center

Hardware Mechanism #2 - PE-local Register File

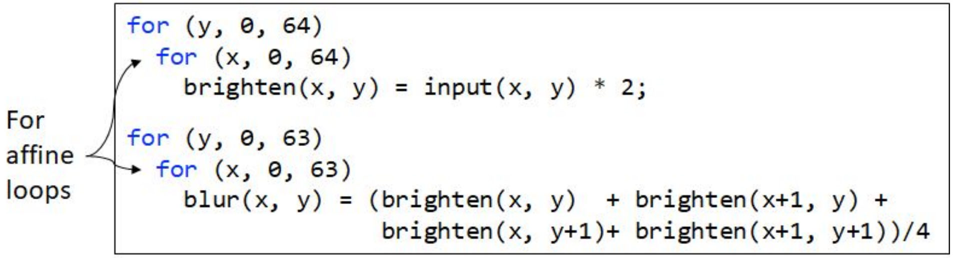
- Mechanism 2: Register file in the PE
  - The lowest level memory hierarchy
- Keep data local to PE for accumulation and low-latency buffering
- Avoid SRAM accesses by exploiting locality



AHA Agile Hardware Center

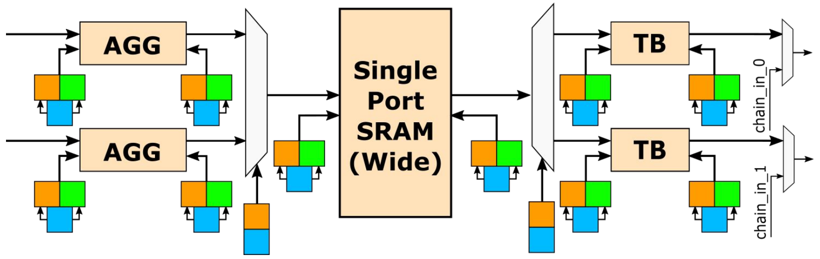
Hardware Mechanism #3 - Unified Framework

So far, the discussion has concerned mainly architecture...what about control?



For affine loops

```
for (y, 0, 64)
  for (x, 0, 64)
    brighten(x, y) = input(x, y) * 2;

for (y, 0, 63)
  for (x, 0, 63)
    blur(x, y) = (brighten(x, y)  + brighten(x+1, y) +
                  brighten(x, y+1)+ brighten(x+1, y+1))/4
```



- Mechanism 3: Unified framework of memory generator
  - Use the same framework for every memory level in the hierarchy
  - Different SRAM organization and addressing complexity

- Compiler analyzes looping structures of the application with given resource constraints
  - Generate configuration registers

AHA Agile Hardware Center

Summary

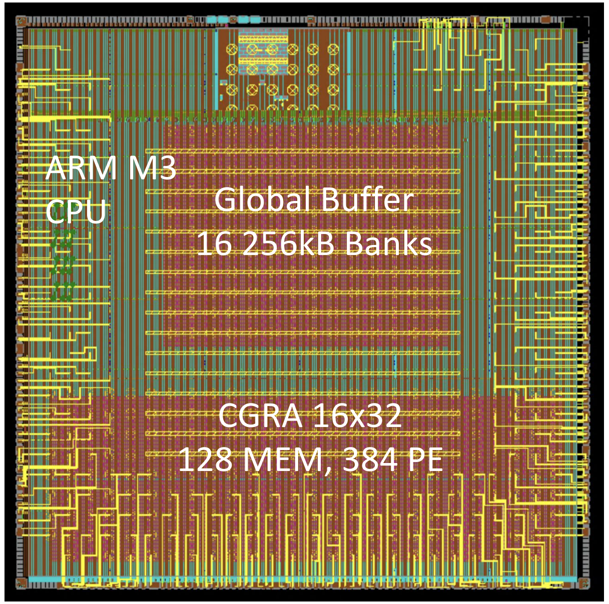
- **Amber** is a prototype for hardware mechanisms for CGRA **virtualization**
    - Multi-kernels in parallel
    - Relocatable bitstream
    - Dynamic partial reconfiguration

- **Amber** has an **energy-efficient** multi-level memory system
    - Wide-fetch, single-port SRAM in MEM tile
    - Register-file in PE tile
    - Unified memory generator framework for each level of memory

Tapeout Methodology

1. Building a chip in the cloud

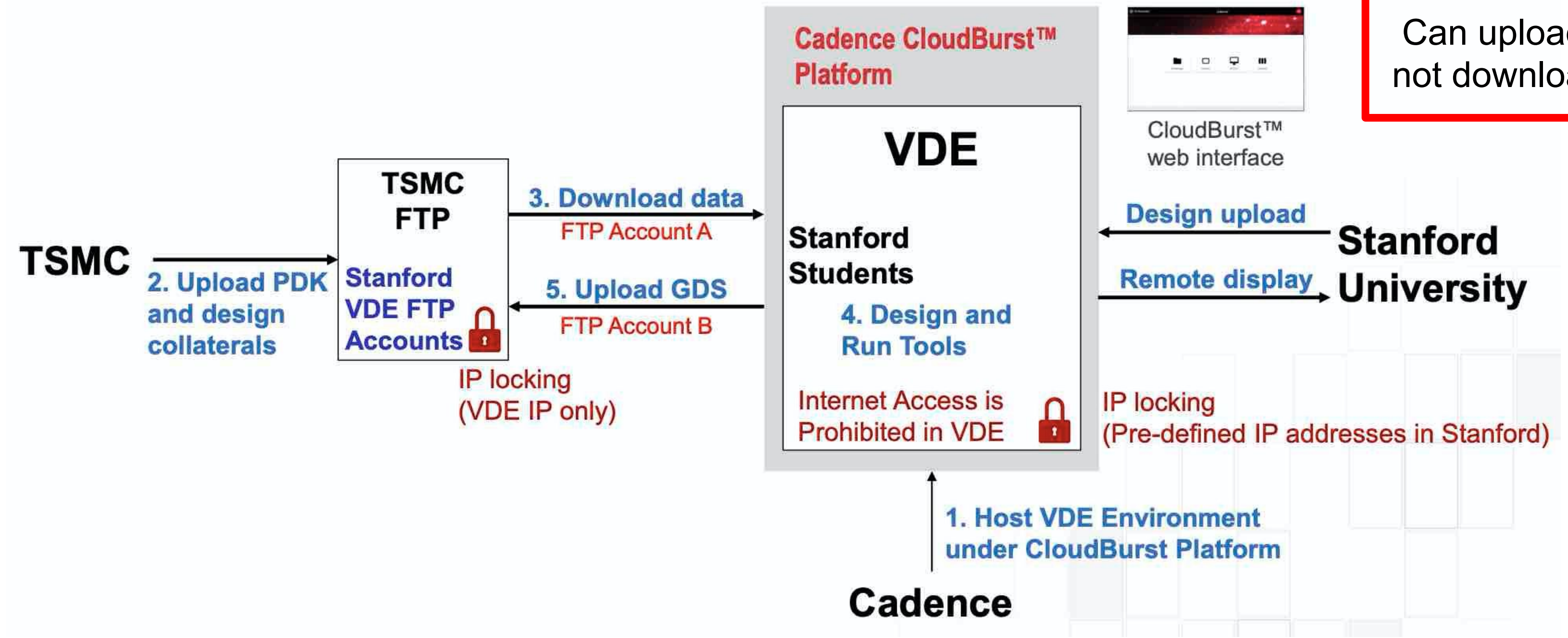
2. Reflecting on a first attempt at agile physical development



AHA
Agile Hardware Center

We built Amber in TSMC Virtual Design Environment (VDE)

- **Vendor**: Prevent accidental leakage of TSMC's technology information
- **Customer**: Cloud EDA


Cadence CloudBurst™ Platform
VDE
Stanford Students
4. Design and Run Tools
Internet Access is Prohibited in VDE
CloudBurst™ web interface
TSMC
2. Upload PDK and design collaterals
TSMC FTP
Stanford VDE FTP Accounts
IP locking (VDE IP only)
3. Download data
FTP Account A
5. Upload GDS
FTP Account B
Design upload
Remote display
Stanford University
IP locking (Pre-defined IP addresses in Stanford)
1. Host VDE Environment under CloudBurst Platform
Cadence

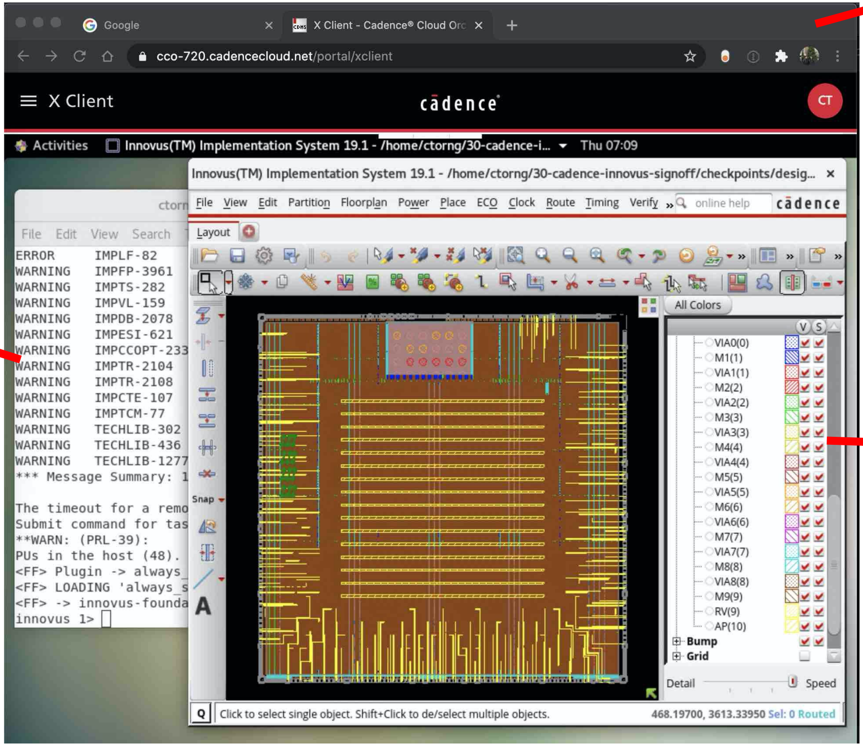
Can upload, not download

We built Amber in TSMC Virtual Design Environment (VDE)

Logging in from a browser

Access to a terminal

Display available for GUI needs


Google
X Client - Cadence® Cloud Orc
cco-720.cadencecloud.net/portal/xclient
X Client
cadence
CT
Activities
Innovus(TM) Implementation System 19.1 - /home/ctorng/30-cadence-i...
Thu 07:09
Innovus(TM) Implementation System 19.1 - /home/ctorng/30-cadence-innovus-signoff/checkpoints/desig...
File View Edit Partition Floorplan Power Place ECO Clock Route Timing Verify
online help
Layout
All Colors
V S
VIA0(0)
M1(1)
VIA1(1)
M2(2)
VIA2(2)
M3(3)
VIA3(3)
M4(4)
VIA4(4)
M5(5)
VIA5(5)
M6(6)
VIA6(6)
M7(7)
VIA7(7)
M8(8)
VIA8(8)
M9(9)
RV(9)
AP(10)
Bump
Grid
Detail
Speed
Snap
ctorn
File Edit View Search

```
ERROR     IMPLF-82
WARNING   IMPFP-3961
WARNING   IMPTS-282
WARNING   IMPVL-159
WARNING   IMPDB-2078
WARNING   IMPESI-621
WARNING   IMPCCOPT-233
WARNING   IMPTR-2104
WARNING   IMPTR-2108
WARNING   IMPCTE-107
WARNING   IMPTCM-77
WARNING   TECHLIB-302
WARNING   TECHLIB-436
WARNING   TECHLIB-1277
*** Message Summary: 1

The timeout for a remo
Submit command for tas
**WARN: (PRL-39):
PUs in the host (48).
<FF> Plugin -> always_
<FF> LOADING 'always_s
<FF> -> innovus-founda
innovus 1>
```

Click to select single object. Shift+Click to de/select multiple objects.   468.19700, 3613.33950 Sel: 0 Routed

AHA Agile Hardware Center

VDE was a success in many respects

From TSMC and Cadence’s perspective

- TSMC protected their sensitive information from accidental leakage
- Cadence has a new platform to offer as a service

From Stanford’s perspective

- We **successfully built a chip** in the cloud
- We had access to all the tools, IP, and compute we needed
- Unique for academics -- We had access to live EDA support (Cadence)

AHA
Agile Hardware Center

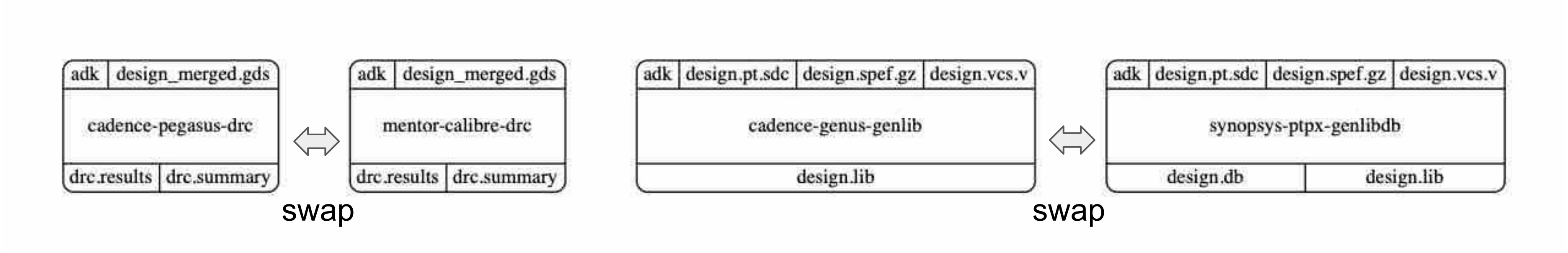
VDE introduced new pain points

Challenge	**Why it can hurt**
Not allowed to extract design and source files from the closed environment (or difficult to)	Losing project history hurts, especially for open-source projects. Difficult to share code with non-VDE projects.
Cloud platform sometimes high latency	Interactive jobs can be very slow
Each VDE instance is a “new system”	Need to spend time getting tools to work again
Single-vendor toolchain (Cadence)	Less flexibility + Team may not have experts
Having to juggle cloud resources vs costs (e.g., making sure big jobs run on big servers)	Overpay for cloud resources or suffer the low performance

AHA
Agile Hardware Center

Our take on the VDE

- Modularity with **mflowgen helped us quickly adapt** between a single-vendor flow (Cadence) and a mixed-vendor flow (Synopsys, Cadence, Mentor)

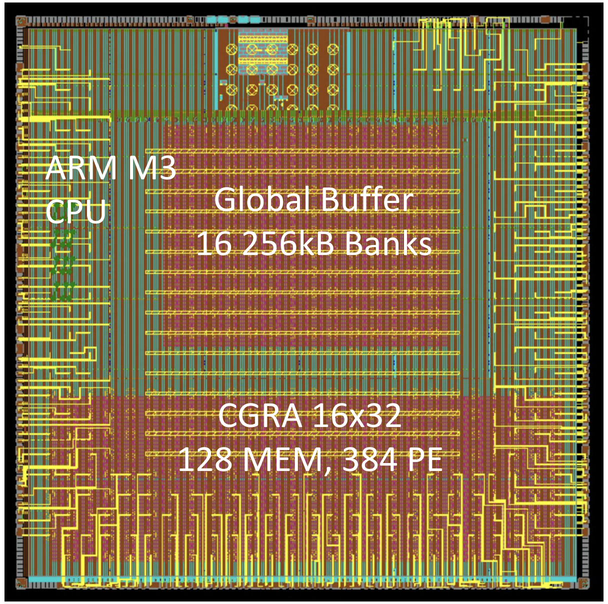


- We maintained version control outside the VDE and **repeatedly uploaded our changes** (only lightweight changes)

- Other challenges decreased productivity (robustness, latency, tools, maintenance), but the **chip still went out successfully**

Tapeout Methodology

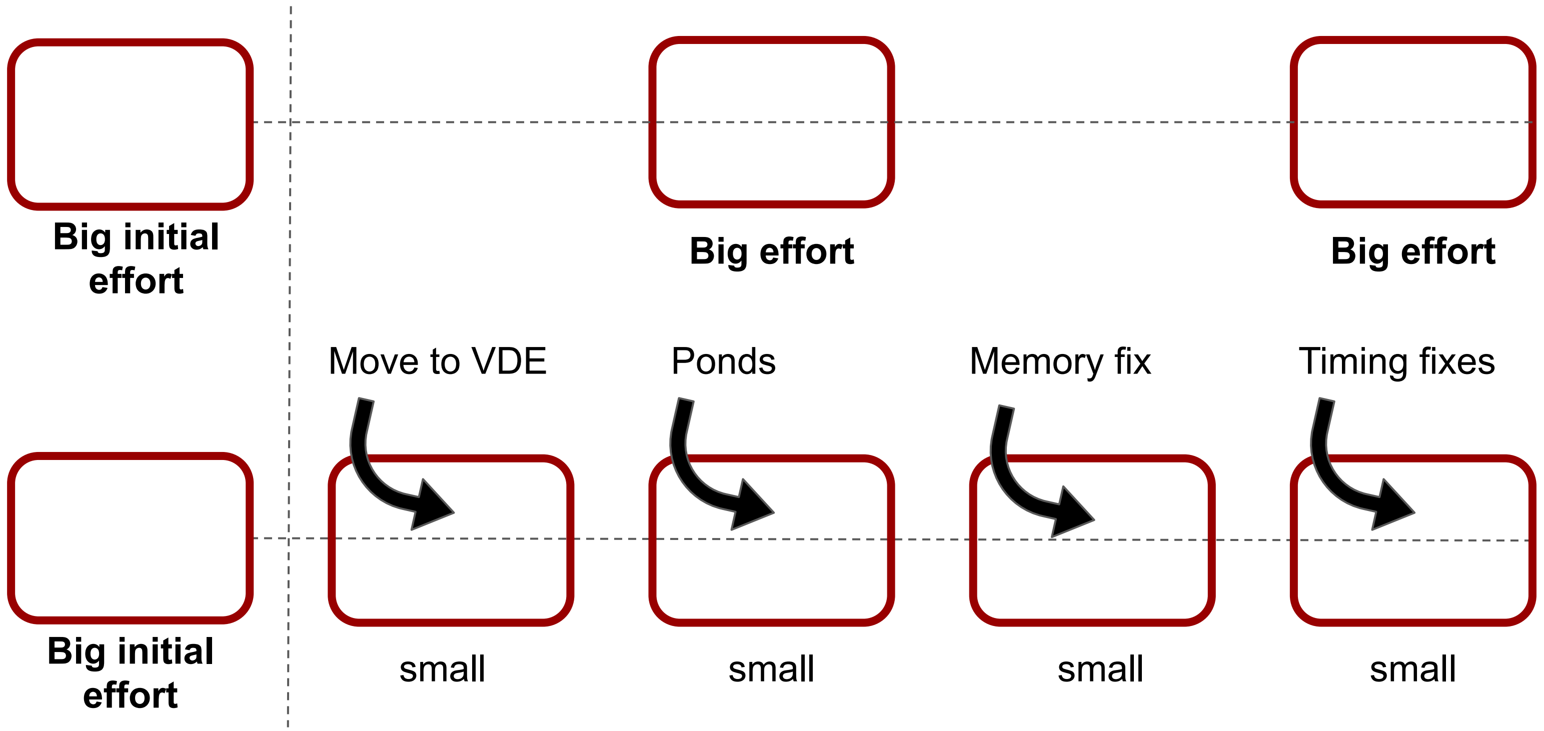
1. Building a chip in the cloud
    - VDE was a success with room for improvement
2. Reflecting on a first attempt at agile physical development
    - Producing a golden tape every week



AHA
Agile Hardware Center

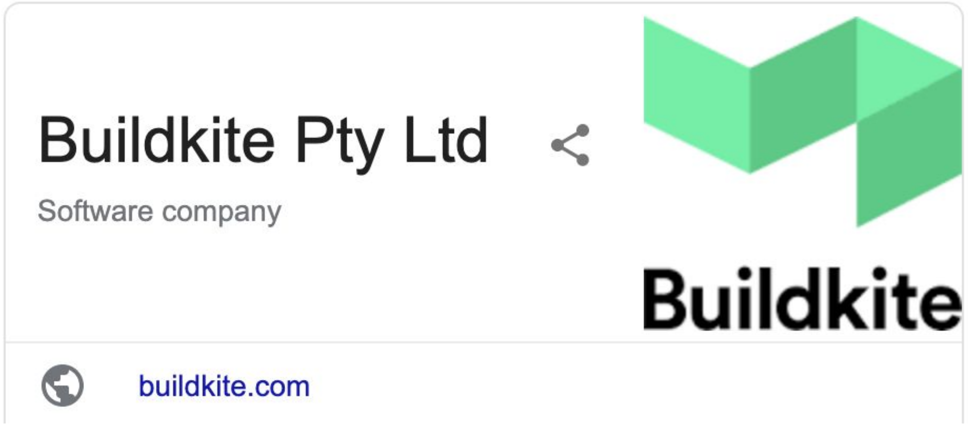
Weekly golden releases instead of “one big release”

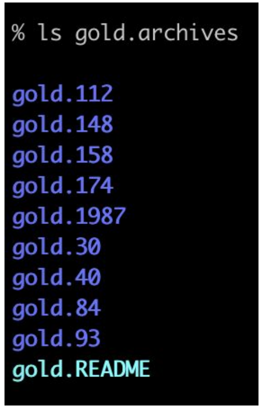
Push away from “**one big release**” and towards software-like **frequent small ones**



Weekly golden releases instead of “one big release”

- We were successful to a large degree...





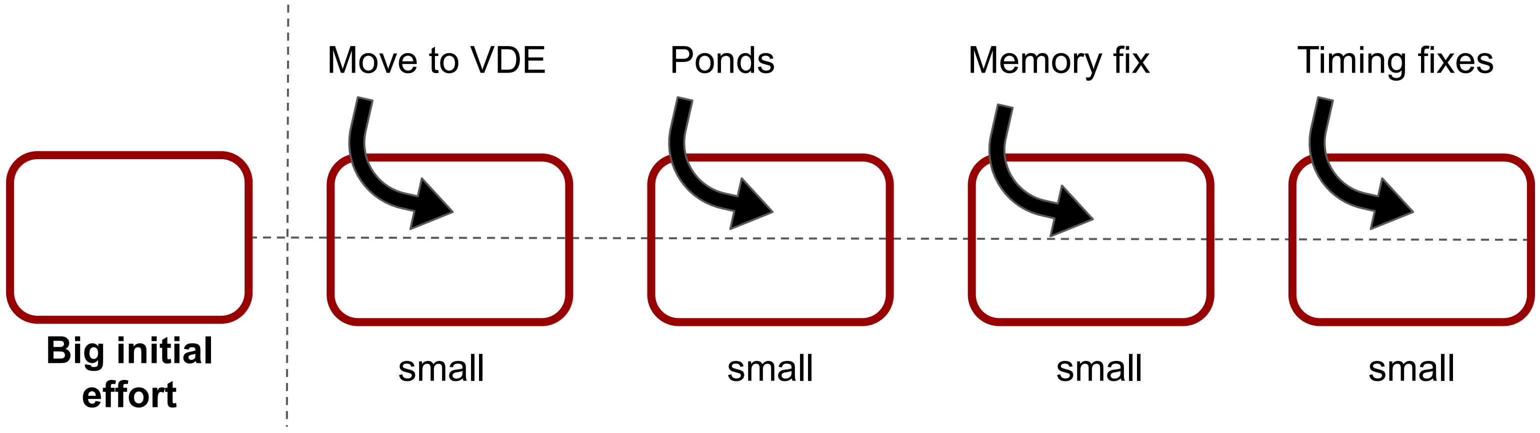
```
% ls gold.archives

gold.112
gold.148
gold.158
gold.174
gold.1987
gold.30
gold.40
gold.84
gold.93
gold.README
```

A history of golden builds throughout design development

- Required **culture change** and also surfaced **fundamental challenges**

AHA
Agile Hardware Center

Agile culture shift -- Biweekly sprints for team focus

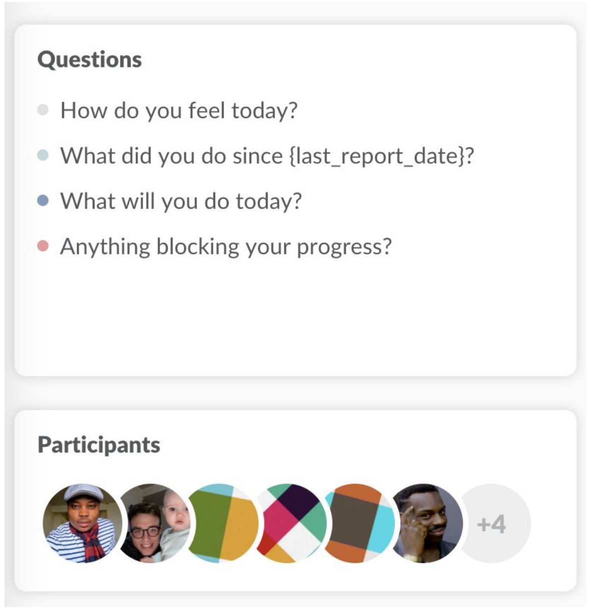


Biweekly sprints helped us convert many unordered parallel efforts into a **progression of team efforts** that “commit” one by one into the chip

AHA
Agile Hardware Center

Agile culture shift -- Adapting Scrums for COVID

Geekbot

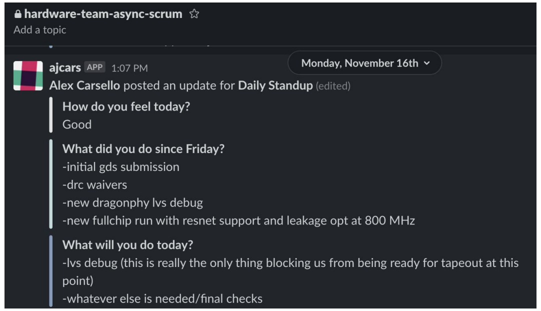


Questions

- How do you feel today?
- What did you do since {last_report_date}?
- What will you do today?
- Anything blocking your progress?

Participants

+4



hardware-team-async-scrum
Add a topic

Monday, November 16th

ajcars APP 1:07 PM
Alex Carsello posted an update for Daily Standup (edited)

How do you feel today?
Good

What did you do since Friday?
-initial gds submission
-drc waivers
-new dragonphy lvs debug
-new fullchip run with resnet support and leakage opt at 800 MHz

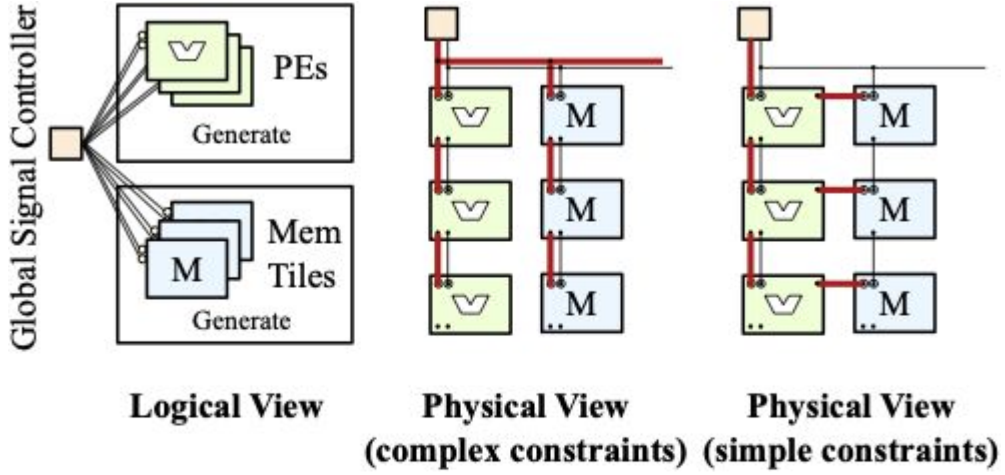
What will you do today?
-lvs debug (this is really the only thing blocking us from being ready for tapeout at this point)
-whatever else is needed/final checks

Slack channel with everyone's responses

Asynchronous scrums completely replaced synchronous ones to great success (e.g., staying in sync, text records, less meeting fatigue)

AHA Agile Hardware Center

Challenge #1 -- Some design approaches are friendlier than others



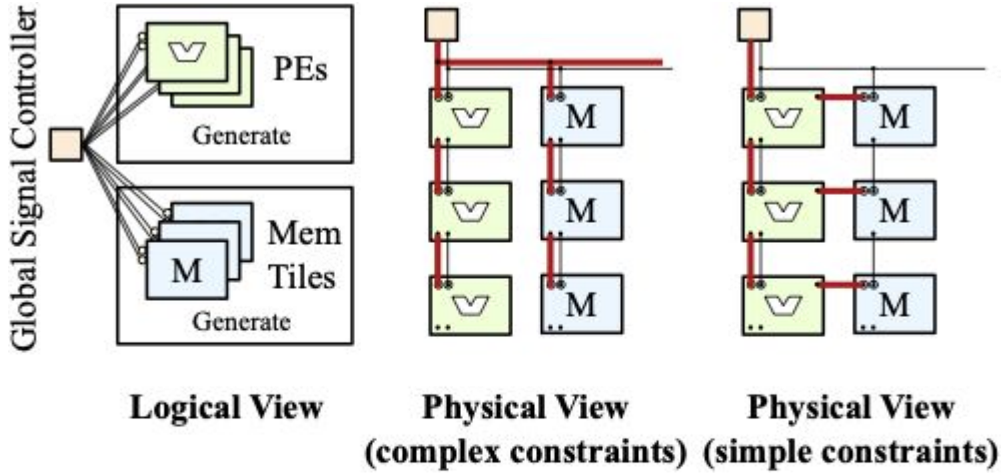
The (left) physical view requires **manual attention to constraints**.

The (right) physical view requires **negligible human attention**

While difficult to spot, certain design decisions which have no impact on logical design can cost weeks of physical iteration.
These need to be eliminated.

AHA
Agile Hardware Center

Challenge #2 -- No single source of truth between physical/logical



Significant design churn when…

- The **logical design updates, but physical collateral is stale** (e.g., constraints)
- The logical design updates **non-functionally** but generates a different netlist which breaks physical scripts

Need a mechanism to better tie together related logical and physical design files

AHA
Agile Hardware Center

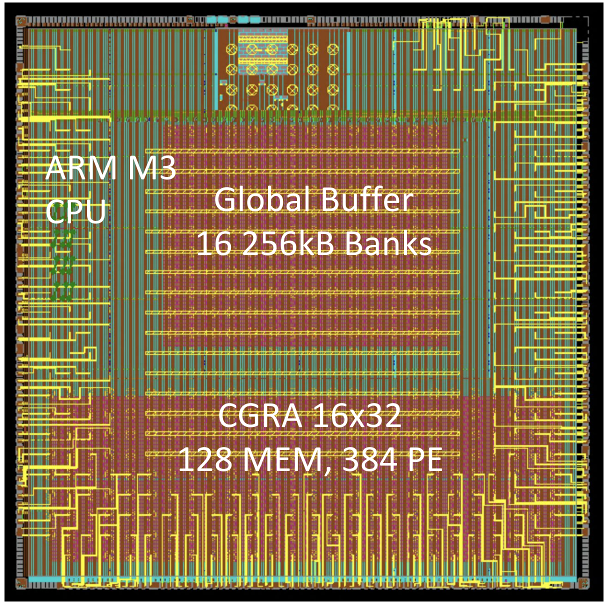
Can we address these two challenges?

- Challenge #1 -- Some design approaches are friendlier than others
- Challenge #2 -- No single source of truth between physical/logical

Summary of Amber SoC tapeout

- Taped out on December 1, 2020
- Garnet PE + **Local register file (ponds)**
- Garnet memory tiles + **Easier to program from the compiler**
- Verified fast reconfiguration, application partitioning, temporal and spatial multiplexing of applications
- Lessons learned in the **cloud**
- Lessons learned for **agile design**

TSMC 16nm
Cadence VDE
5mm x 5mm



AHA
Agile Hardware Center