ACM/IEEE Lab Tour

Henry Dietz & Paul Eberhart

108 Marksbury, 6PM March 20, 2023

University of Kentucky
Electrical & Computer Engineering

Aggregate.Org UNBRIDLED COMPUTING

University of Kentucky

Primary Research Motivation

Make the hardware and software components of a computing system work better together

- To **improve performance**
- To **gain new abilities**

Our Name

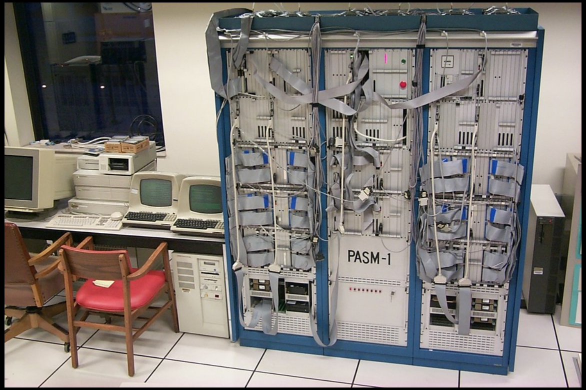
- Originally **CARP**: Compiler-oriented Architecture Research at Purdue
- Became **PAPERS**: Purdue's Adapter for Parallel Execution and Rapid Synchronization
- Became Aggregate.Org UNBRIDLED COMPUTING and in UK KAOS Compilers Hardware Architectures Operating Systems

What Is A Supercomputer?

A computer that can solve big problems *and* can scale to solve bigger problems

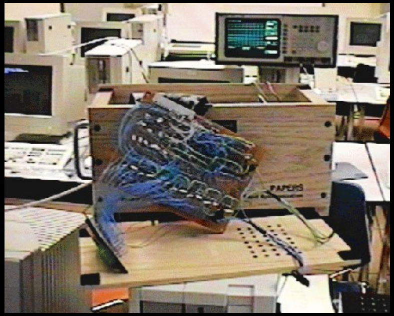
- Mostly about **parallel processing** – divide work into pieces that execute simultaneously
- Need not be huge, expensive, etc.

Coordination



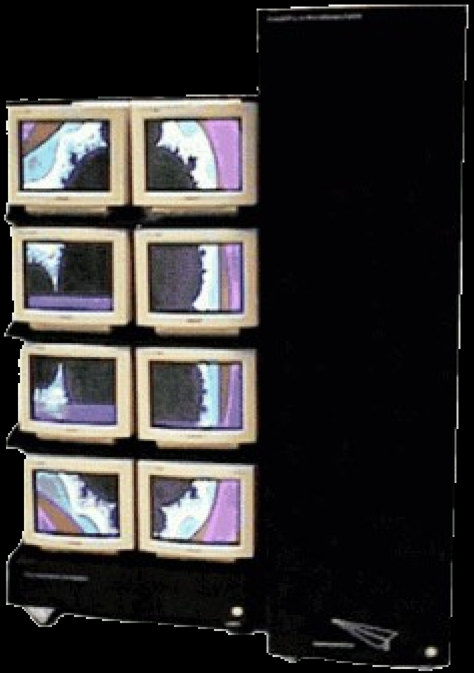
- Parallel algorithms want cheap sampling of global state: **barrier synchronization** hardware
- The **aggregate function network** (**AFN**) model combines barrier synchronization with collective operations performed within the network

Cluster Supercomputers



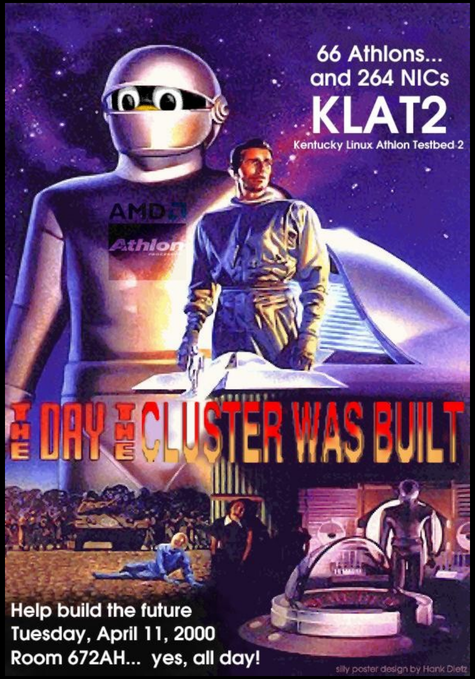
- Prototype PAPERS AFN using standard hardware interfaces

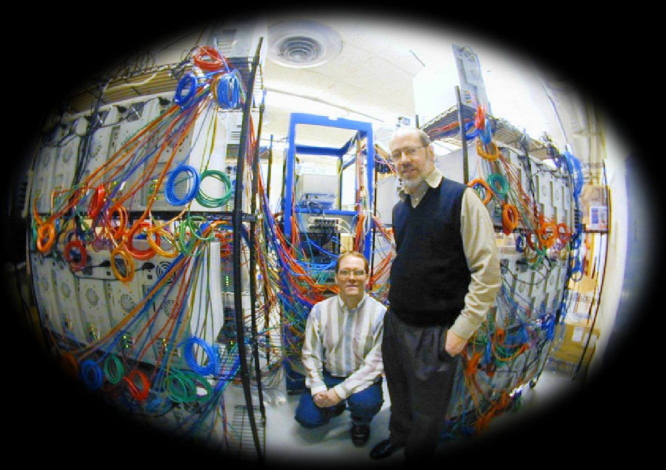
- 1^{st} was in 2/94; by :
  1^{st} 386 & 486 Linux PC clusters
  1^{st} IBM PowerPC AIX cluster
  1^{st} DEC Alpha OSF cluster
  1^{st} **sync'd audio/video walls**

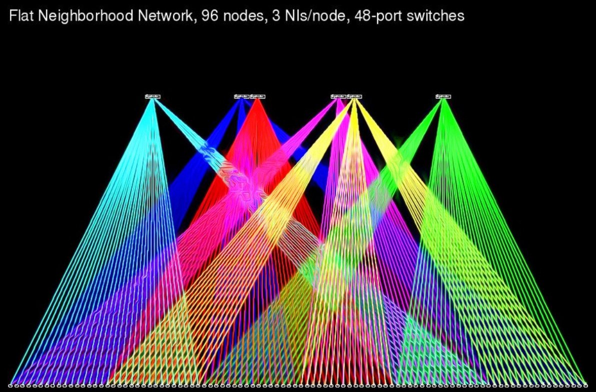


Flat Neighborhood Networks

KLAT2 set various world records, won Computerworld & Bell awards, Using 1st **GA-evolved network**

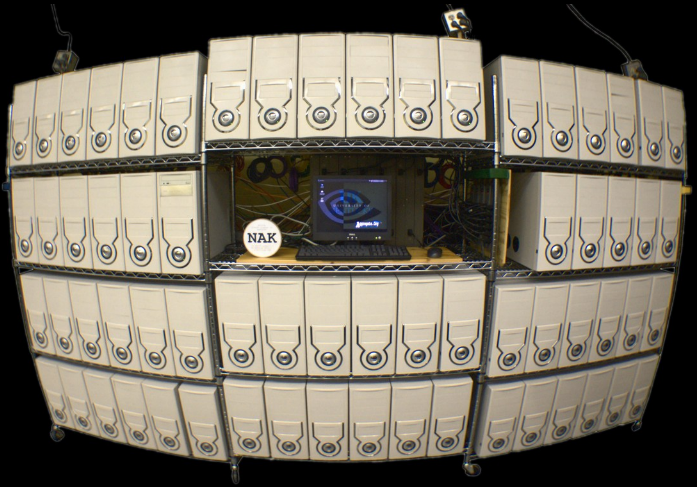






SIMDish Stuff

- Single Instruction, Multiple Data
- **SWAR**: SIMD within a register
- **MOG**: MIMD on **GPU**



The Punchline

- Cost per GFLOPS (1 Billion {+,*} per second):
  1992: MasPar MP1 $1,000,000 / GFLOPS
  2000: KLAT2 $650 / GFLOPS
  2003: KASY0 $84 / GFLOPS
  2010: NAK $0.65 / GFLOPS
  …
- There are now many **HUGE** clusters!

The Punchline

- **There are now many HUGE clusters!**
- 108A Marksbury has 168kW, 30 tons cooling, heats half the Marksbury building, and could not power 1 rack out of thousands in a large system!
- It's really all about power / computation

LCPC 2017:
How Low Can You Go?

- Now it's all about power / computation
- Work only on active bits (bit-serial)
- Aggressive gate-level optimization
- Potential exponential benefit from Quantum?

Reduce gates / word operation

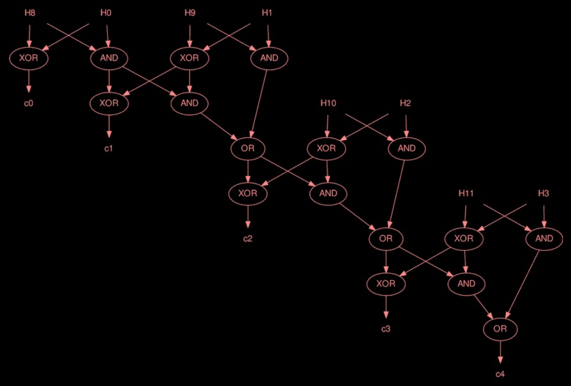
```
int a,b,c; c=a+b;
```

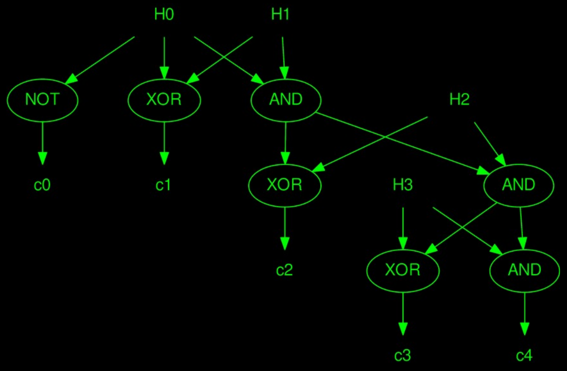
- Fast 32-bit Carry Lookahead:
  ~645 gate actions, ~12 gate delays
- 32-bit Ripple Carry, get *throughput* by **SIMD**:
  ~153 gate actions, ~91 gate delays (3 per FA)

Gate-level optimization

```
int:4 a,b; int:5 c; b=1; c=a+b;
```

- 17 gates becomes 7 when optimized





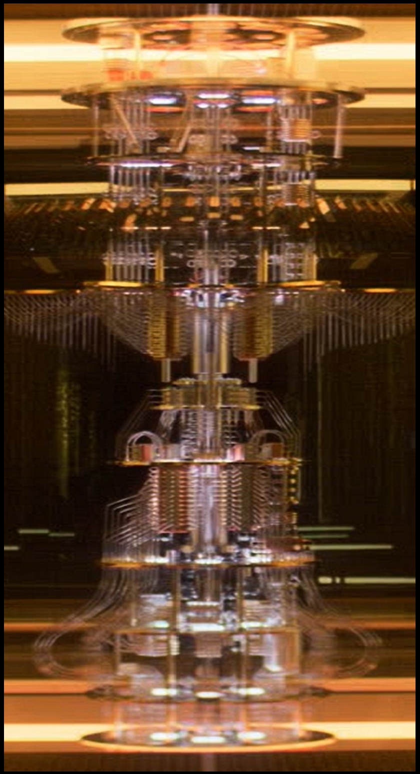
Gate-Level Optimization

```
int:8 a, b, c;
a = (c * c) ^ 70;
a = ((a >> 1) & 1);
a = b + (c * b) + a;
a = a + ~(b * (c + 1));
```

- About 206,669 gates unoptimized
- Optimized, it's just `a = 0;`

Bits are better than words!

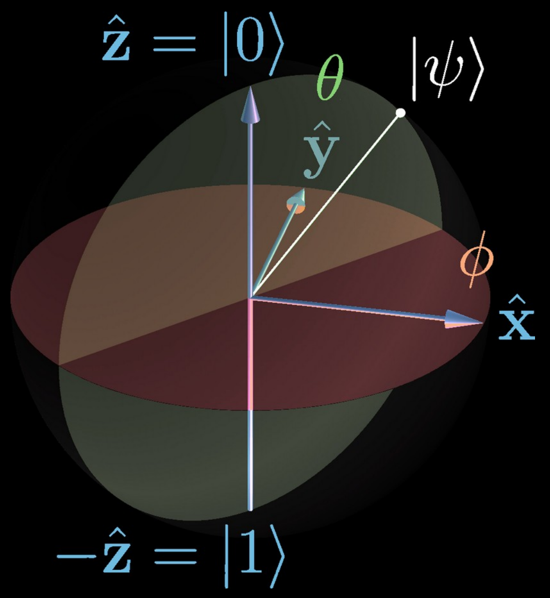
Quantum Computers



Quantum Computing

- Superposition: 1 qubit, all values
- Entanglement: *e* qubits, 2^e values
  - Exponentially less memory
  - Exponentially fewer gate ops

- Limited coherence, no cloning, only reversible logic gates, ...



Parallel Bit Pattern Computing

- Compression for entangled superposition / SIMD
  - Up to exponential reduction in storage, gate ops
- Avoids **major quantum problems**:
  - Forever coherent, error free
  - Cloning: fanout, non-destructive measurement
  - Use any gates, not just reversible logic
  - We know how to build scalable hardware

Parallel Bit Patterns (2^e bits)

- 2^e is **`nproc`**...

```
10101010 10101010 10101010 10101010
11001100 11001100 11001100 11001100
11110000 11110000 11110000 11110000
11111111 00000000 11111111 00000000
11111111 11111111 00000000 00000000
```

- For **`nproc`**=32, **`iproc`** is:
  (apparently 5x8x4=160 bits to store)

```
chunk(2) chunk(2) chunk(2) chunk(2)
chunk(3) chunk(3) chunk(3) chunk(3)
chunk(4) chunk(4) chunk(4) chunk(4)
chunk(1) chunk(0) chunk(1) chunk(0)
chunk(1) chunk(1) chunk(0) chunk(0)
```

- For 8-bit chunks, this is:
  (only 5 chunks used, so **just 5x8=40 bits stored**)

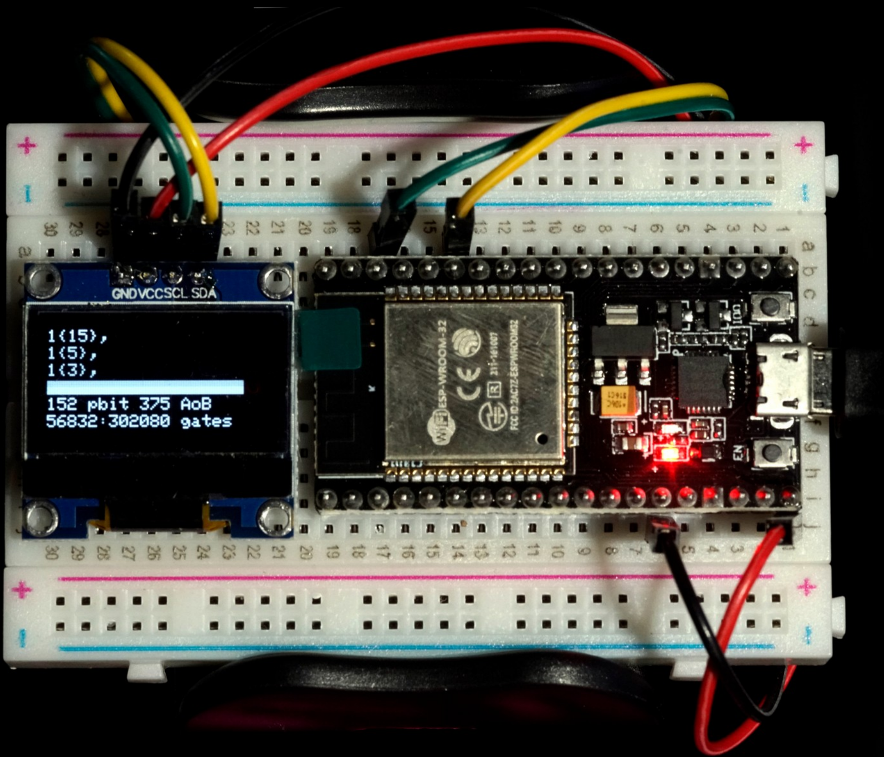
```
chunk(1) chunk(1) chunk(1) chunk(1)
```

- To add 1 to **`iproc`**, we add:
  (**only chunk operations *with unique operands* happen**)

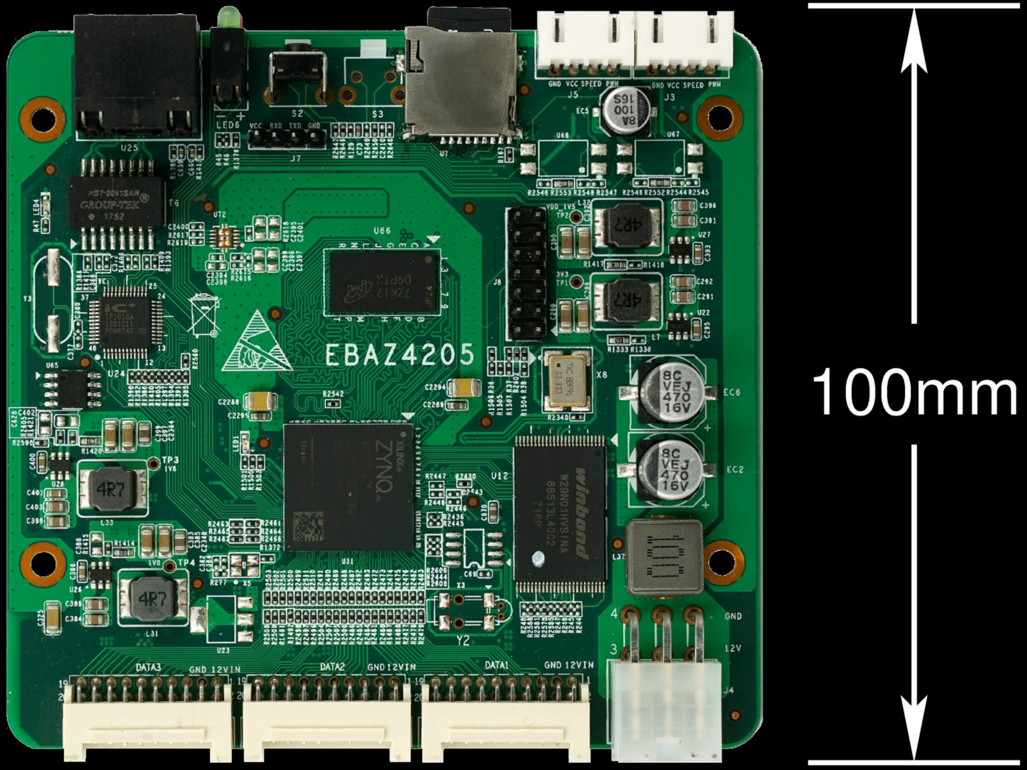
Semiprime Factoring

```
// Semiprime factor to 5-bit primes:
// 2,3,5,7,11,13,17,19,23,29,31
int a = 11*29;
pint b = 0; b = b.Had(5,0); // 1st factor
pint c = 0; c = c.Had(5,5); // 2nd factor
pint d = b * c;             // multiply 'em
pint e = (d == a);          // which gave a?
pint f = e * b;             // 0 non-factors
Print(f);
```

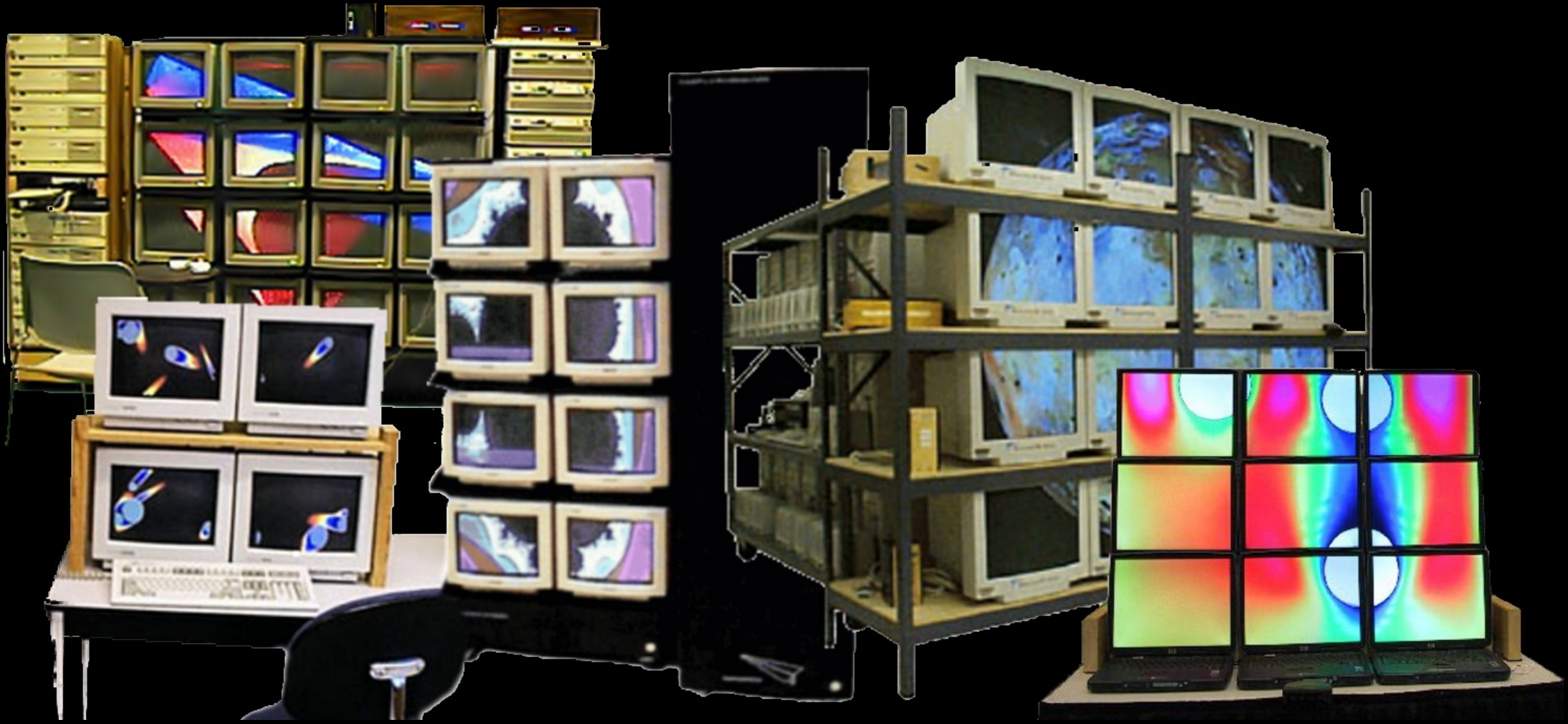
10-way, 1024 pbits… under $10



Our Next Supercomputer



New Abilities for Supers

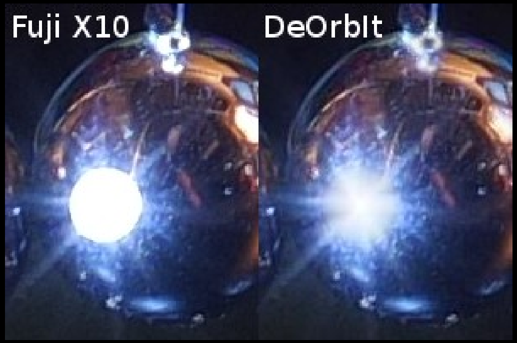


Computational Photography

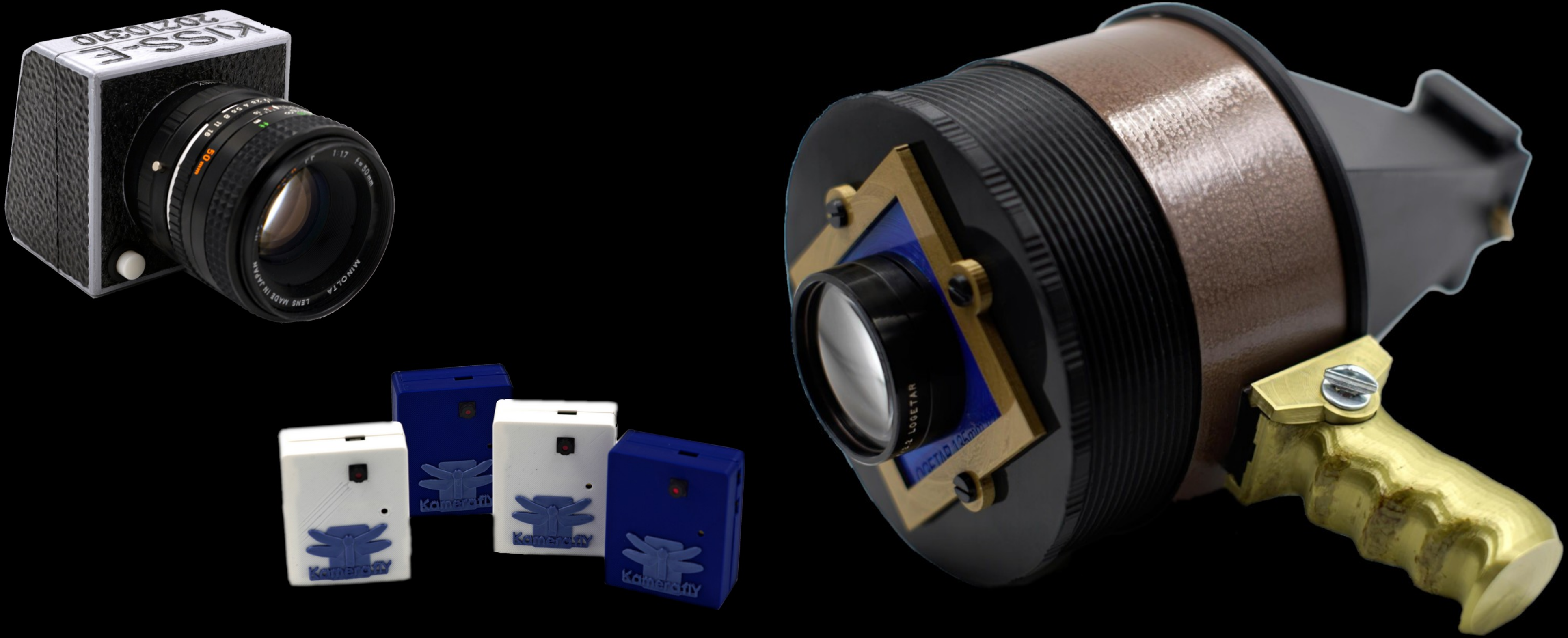
Cameras as computing systems;
using computation to enhance camera abilities
and / or to process the data captured

- New camera / sensor / processing models
- Intelligent computer control of capture
- Detection / manipulation of image properties

Credible repair



Custom cameras



Design For Manufacturability

Design product to be easy to make.

- E.g., FFD 3D printers don’t like spans
- If an object model is a parametric program, this is basically an optimizing compiler...

Design For Manufacturability

- Our systems viewpoint branches to 3 areas:
  - Parallel supercomputing
  - Computational photography
  - Making technologies

- Even undergrads can get involved in Summer research, projects, courses, etc.

Spring 2023: TR 12:30-1:45 EE599-001/EE699-001

Programmable Cameras and IoT

This course will start by introducing the basic principles of photography and the details of how digital cameras work. However, cameras are no longer just about photography; they are *sensors in embedded computing systems* that can serve a wide range of applications. For example, using **CHDK**, it is trivial to program a **Canon PowerShot** camera to serve as a non-contact tape measure. The course will use CHDK cameras to explain how camera internals work and students will get hands-on experience using and programming these cameras. Cameras are also now cheap sensors for use within Internet of Things (IoT) devices. An **ESP32-CAM** IoT module that costs under $10 includes a 2MP camera and can be programmed for tasks as diverse as wirelessly serving live video via an HTML browser interface to unlocking a door when a person's face is recognized. We will discuss IoT devices in general and use of the ESP32-CAM and its OV2640 camera in particular. Students will implement simple IoT projects using the ESP32-CAM via the Arduino programming environment.

Prerequisites: Familiarity with C/C++ programming. No background with photography is required.