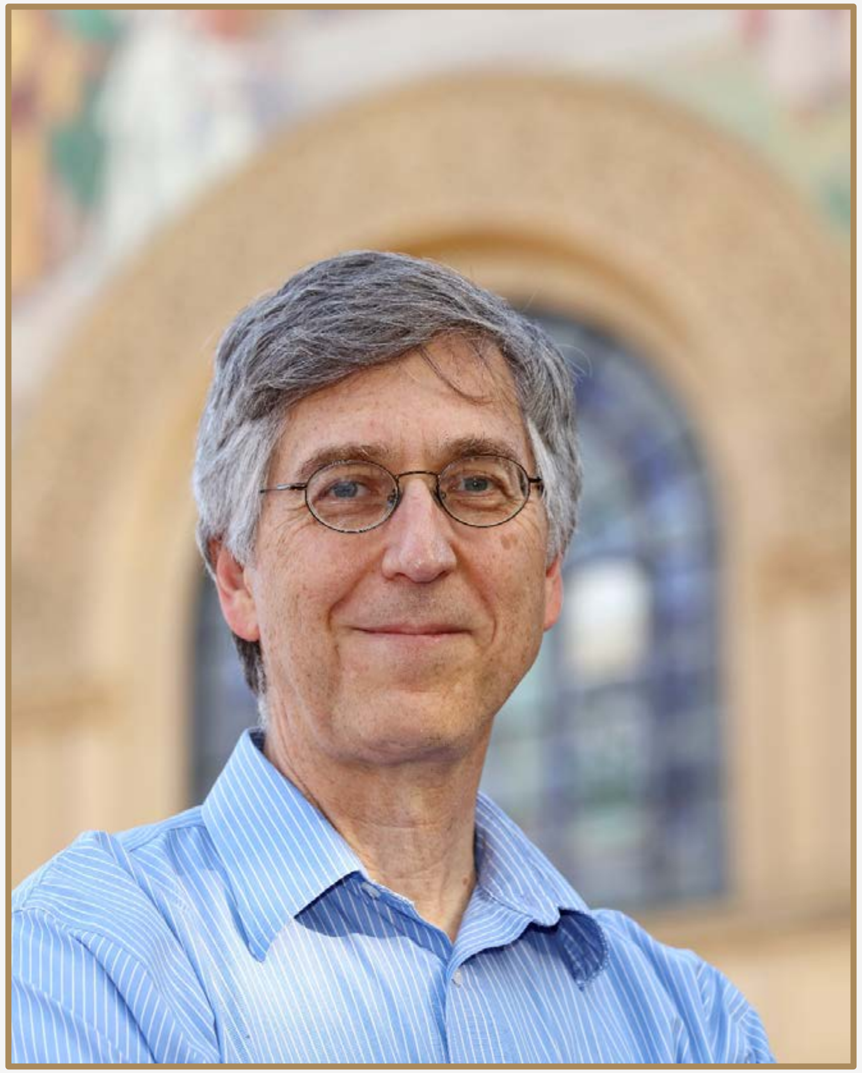


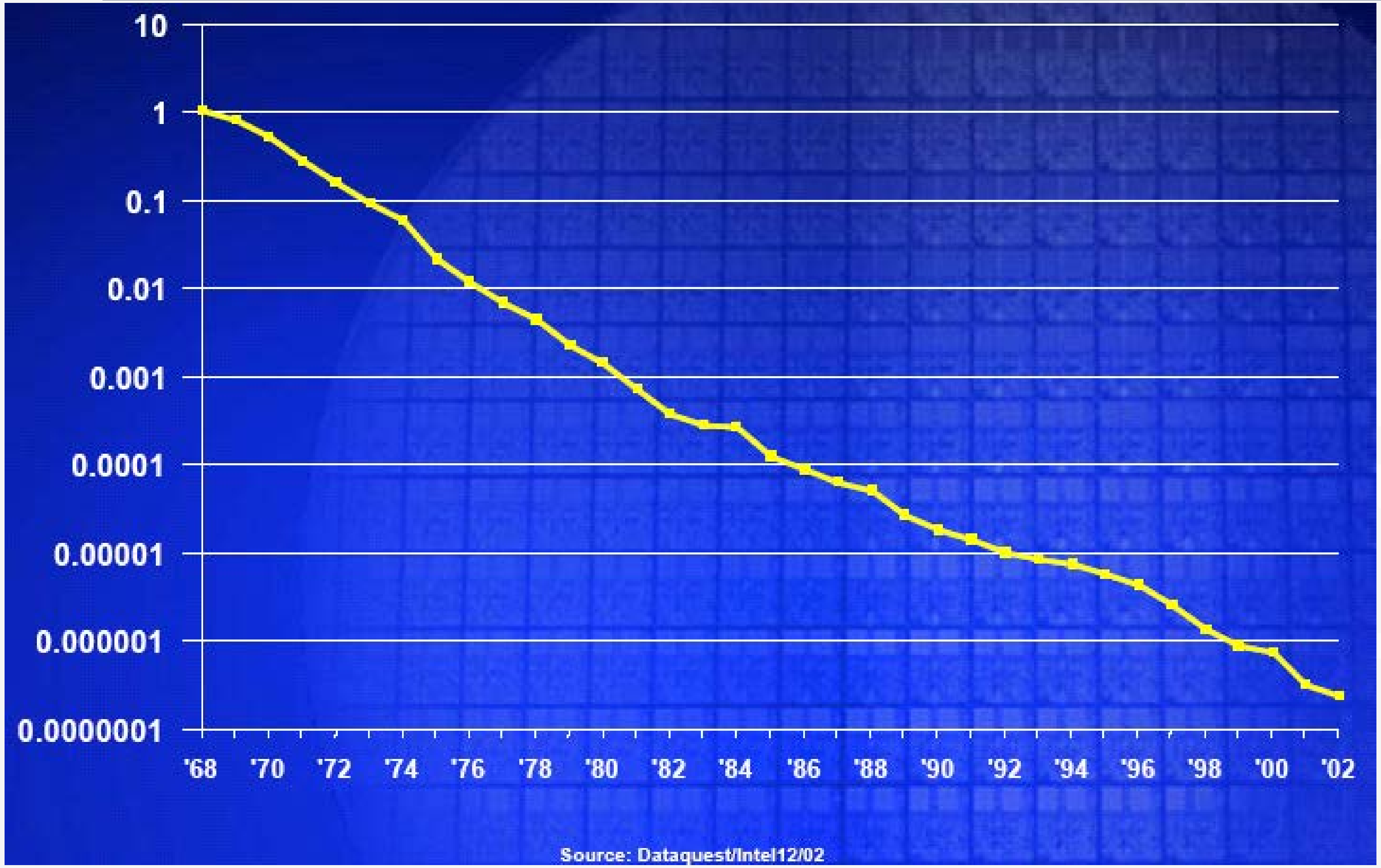
MARK HOROWITZ

STANFORD UNIVERSITY

AHA! – VISUAL COMPUTING

This research was performed with funding from the Defense Advanced Research Projects Agency (DARPA) Domain Specific System on a Chip (DSSoC) Program under the direction of Dr. Thomas Rondeau. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. U.S. Government.

SCALING PROVIDED A GREAT RIDE



No Exponential is Forever...but We Can Delay 'Forever', Moore ISSCC 2002

For a 2x scaling

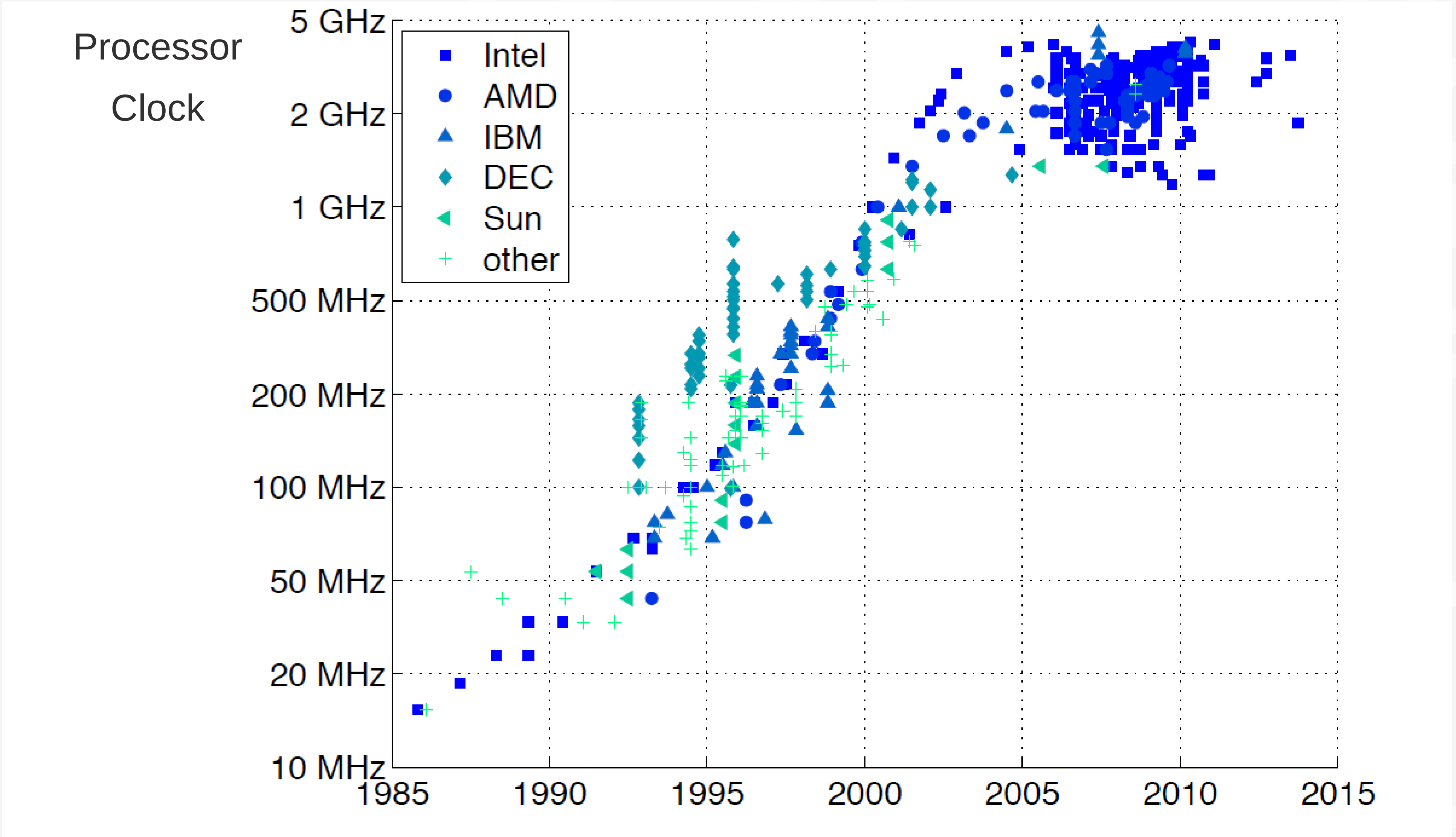
Get 4x more gates,

Gates get 2x faster,

Energy decrease 8x

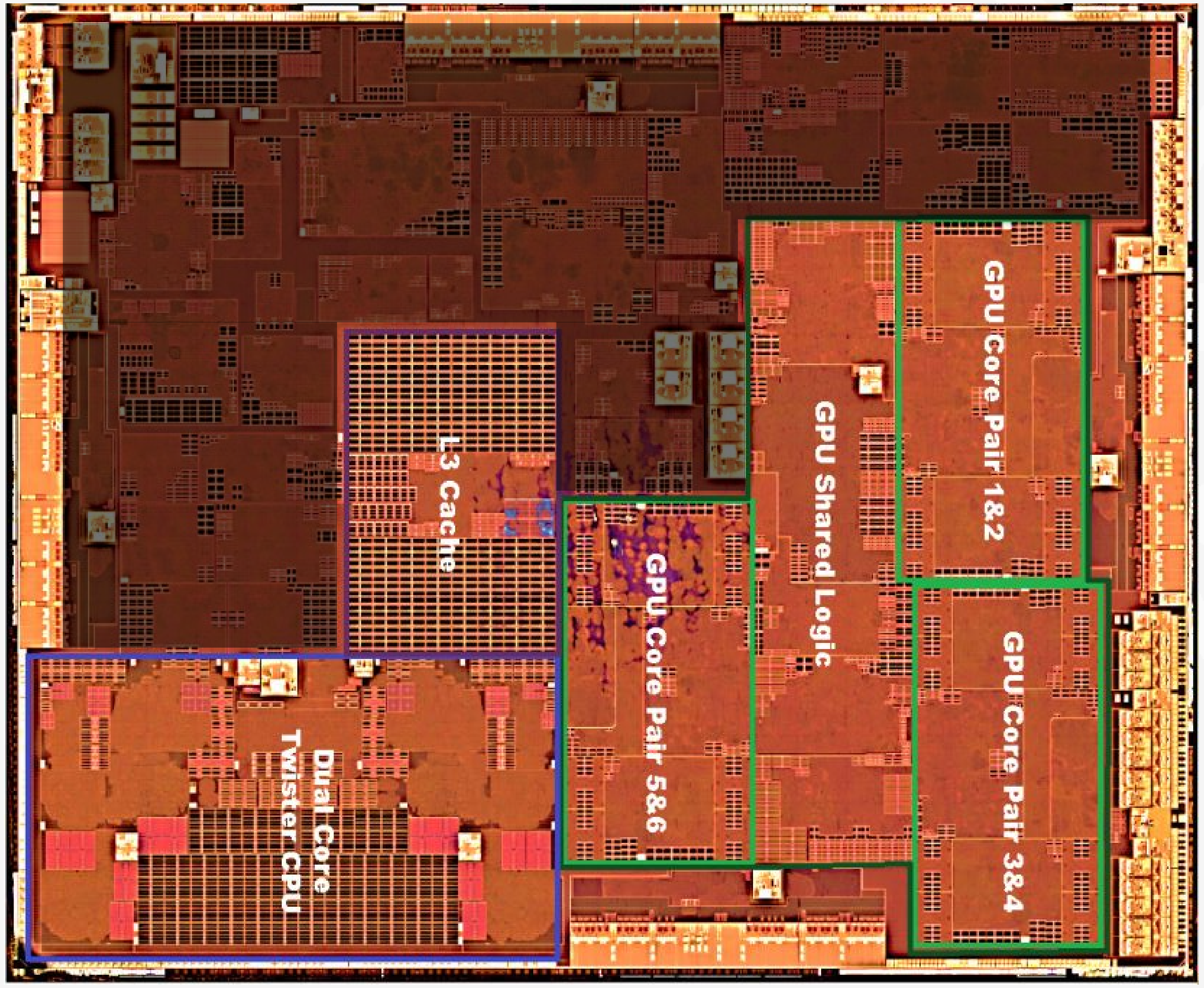
Dennard, JSSC,
pp. 256-268, Oct. 1974

HOUSTON, WE HAVE A PROBLEM



http://cpudb.stanford.edu/

TO CONTINUE TO SCALE PERFORMANCE



Apple’s A9 (2015)

https://www.anandtech.com/show/9686/the-apple-iphone-6s-and-iphone-6s-plus-review/3

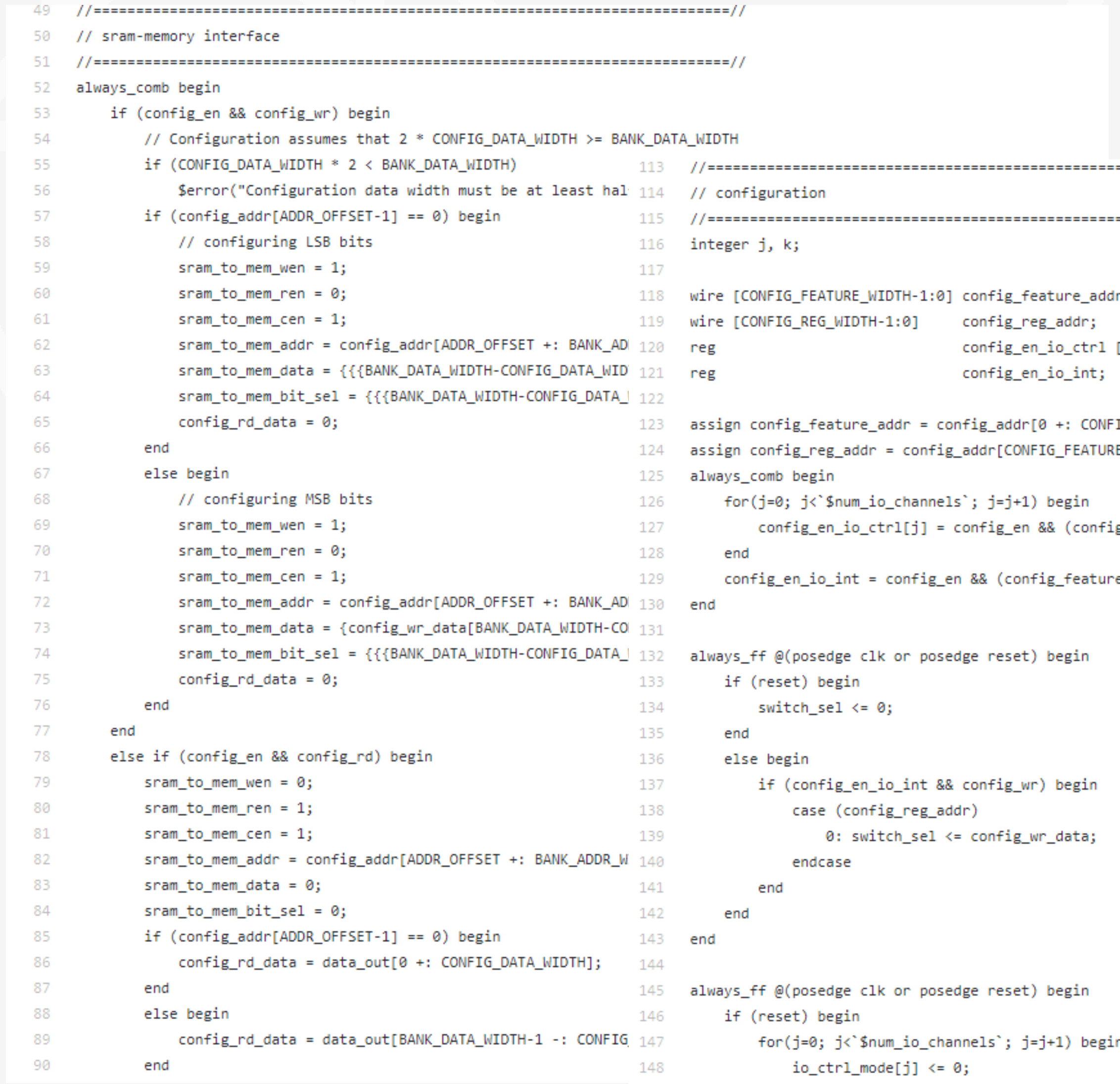
HOW TO CREATE THESE ACCELERATORS?

- **Study application**

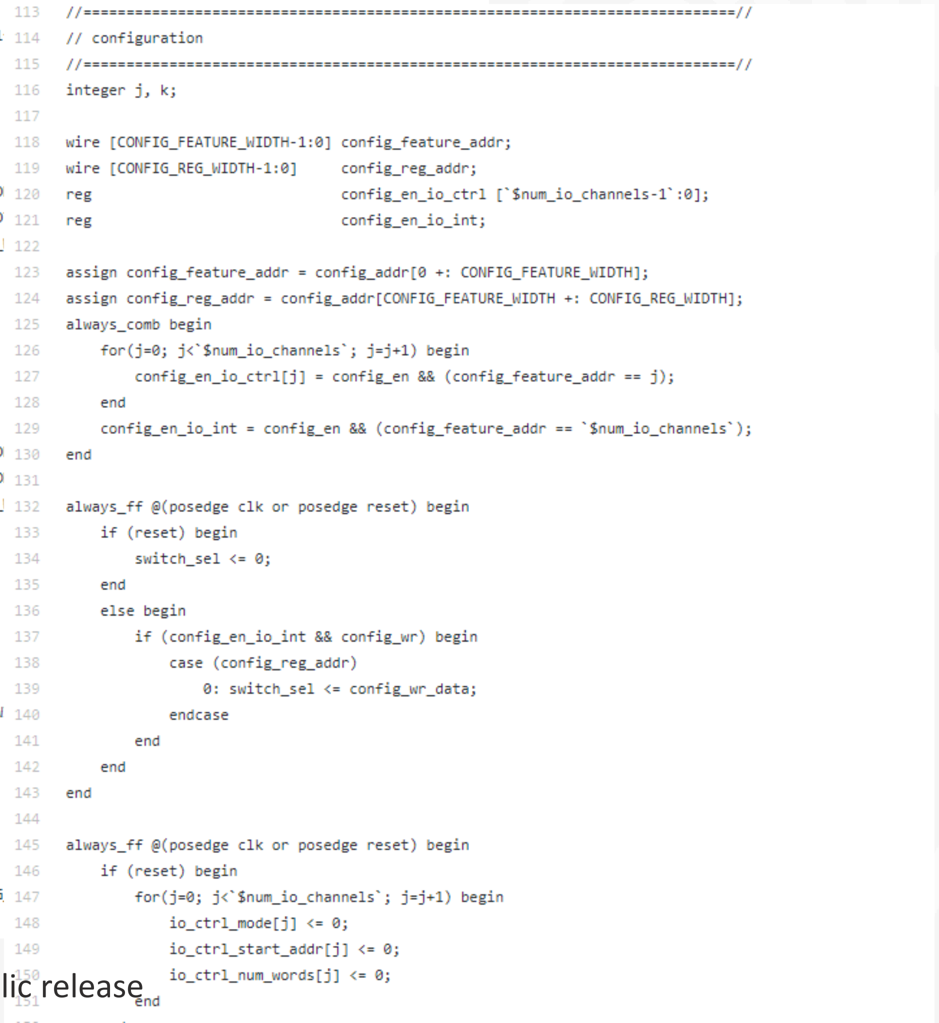
HOW TO CREATE THESE ACCELERATORS?

- Study application

- **Design hardware**



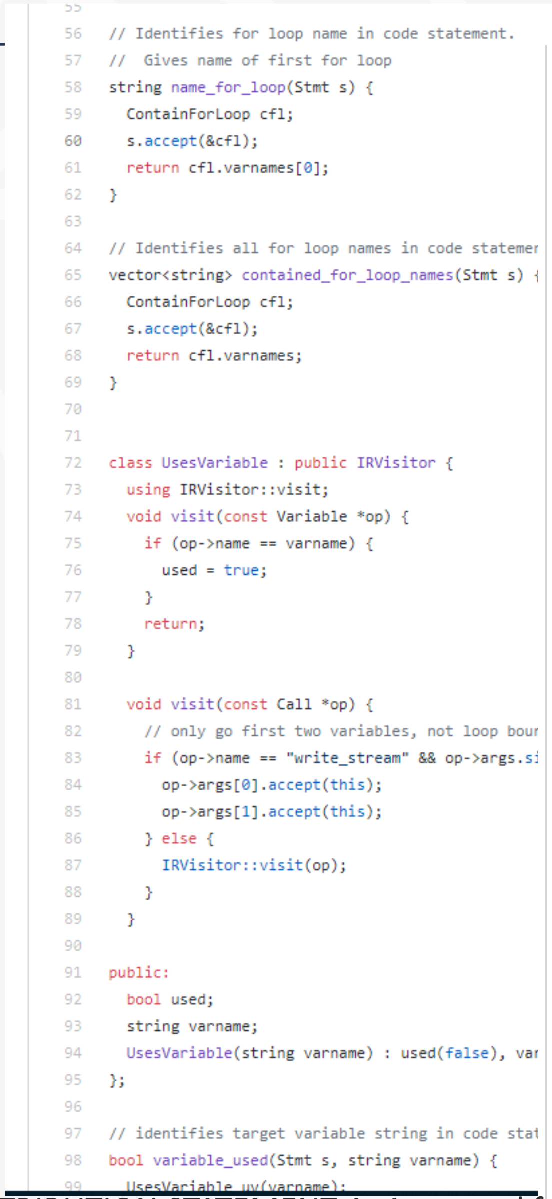
```
//===========================================================================//
// sram-memory interface
//===========================================================================//
always_comb begin
    if (config_en && config_wr) begin
        // Configuration assumes that 2 * CONFIG_DATA_WIDTH >= BANK_DATA_WIDTH
        if (CONFIG_DATA_WIDTH * 2 < BANK_DATA_WIDTH)
            $error("Configuration data width must be at least hal
        if (config_addr[ADDR_OFFSET-1] == 0) begin
            // configuring LSB bits
            sram_to_mem_wen = 1;
            sram_to_mem_ren = 0;
            sram_to_mem_cen = 1;
            sram_to_mem_addr = config_addr[ADDR_OFFSET +: BANK_AD
            sram_to_mem_data = {{{BANK_DATA_WIDTH-CONFIG_DATA_WID
            sram_to_mem_bit_sel = {{{BANK_DATA_WIDTH-CONFIG_DATA_
            config_rd_data = 0;
        end
        else begin
            // configuring MSB bits
            sram_to_mem_wen = 1;
            sram_to_mem_ren = 0;
            sram_to_mem_cen = 1;
            sram_to_mem_addr = config_addr[ADDR_OFFSET +: BANK_AD
            sram_to_mem_data = {config_wr_data[BANK_DATA_WIDTH-CO
            sram_to_mem_bit_sel = {{{BANK_DATA_WIDTH-CONFIG_DATA_
            config_rd_data = 0;
        end
    end
    else if (config_en && config_rd) begin
        sram_to_mem_wen = 0;
        sram_to_mem_ren = 1;
        sram_to_mem_cen = 1;
        sram_to_mem_addr = config_addr[ADDR_OFFSET +: BANK_ADDR_W
        sram_to_mem_data = 0;
        sram_to_mem_bit_sel = 0;
        if (config_addr[ADDR_OFFSET-1] == 0) begin
            config_rd_data = data_out[0 +: CONFIG_DATA_WIDTH];
        end
        else begin
            config_rd_data = data_out[BANK_DATA_WIDTH-1 -: CONFIG
        end
```



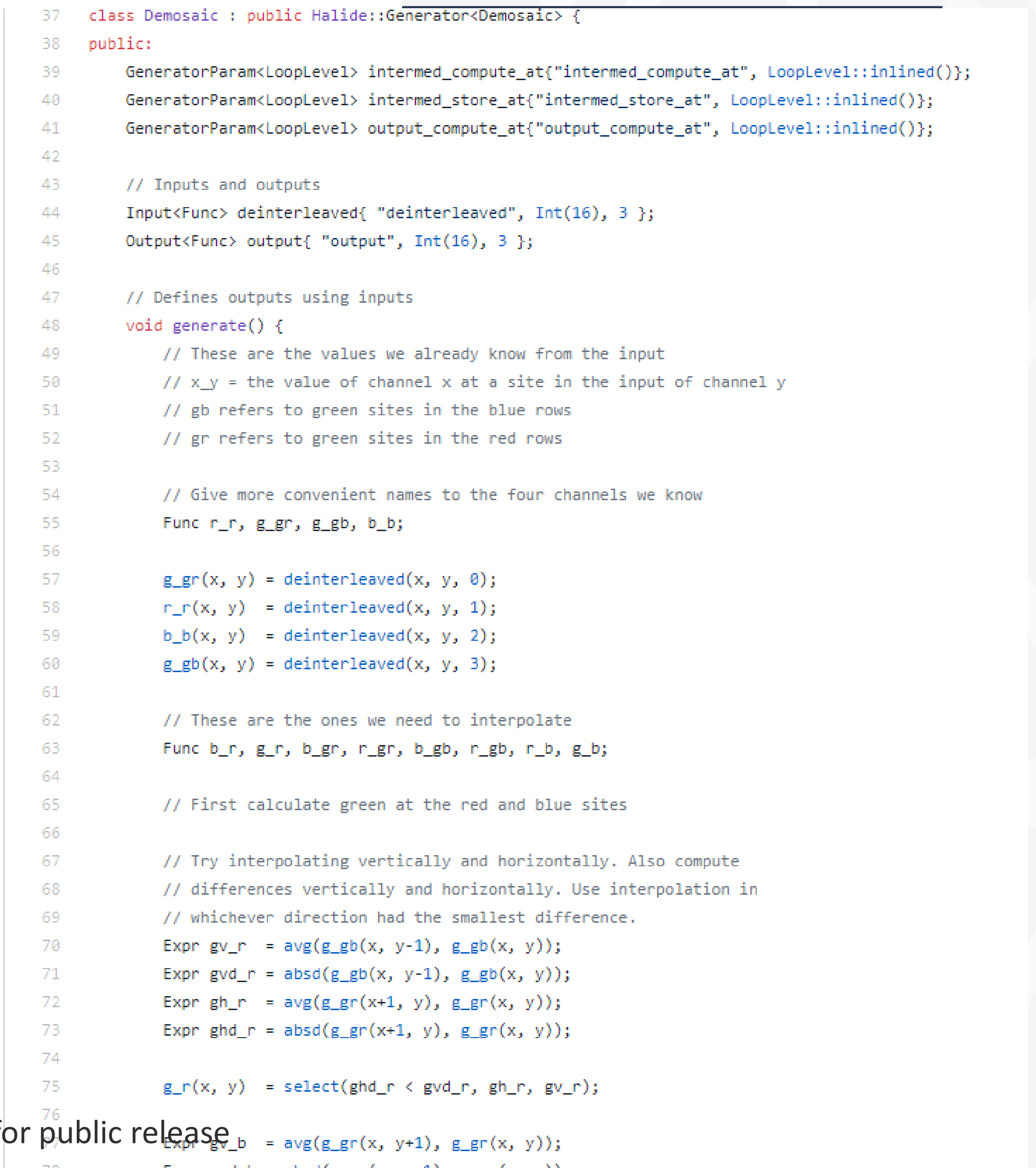
```
//===========================================================================//
// configuration
//===========================================================================//
integer j, k;

wire [CONFIG_FEATURE_WIDTH-1:0] config_feature_addr;
wire [CONFIG_REG_WIDTH-1:0]     config_reg_addr;
reg                             config_en_io_ctrl [`$num_io_channels-1`:0];
reg                             config_en_io_int;

assign config_feature_addr = config_addr[0 +: CONFIG_FEATURE_WIDTH];
assign config_reg_addr = config_addr[CONFIG_FEATURE_WIDTH +: CONFIG_REG_WIDTH];
always_comb begin
    for(j=0; j<`$num_io_channels`; j=j+1) begin
        config_en_io_ctrl[j] = config_en && (config_feature_addr == j);
    end
    config_en_io_int = config_en && (config_feature_addr == `$num_io_channels`);
end

always_ff @(posedge clk or posedge reset) begin
    if (reset) begin
        switch_sel <= 0;
    end
    else begin
        if (config_en_io_int && config_wr) begin
            case (config_reg_addr)
                0: switch_sel <= config_wr_data;
            endcase
        end
    end
end

always_ff @(posedge clk or posedge reset) begin
    if (reset) begin
        for(j=0; j<`$num_io_channels`; j=j+1) begin
            io_ctrl_mode[j] <= 0;
            io_ctrl_start_addr[j] <= 0;
            io_ctrl_num_words[j] <= 0;
        end
```

HOW TO CREATE THESE ACCELERATORS?

- Study application
- Design hardware
- **Write software**



```
// Identifies for loop name in code statement.
//  Gives name of first for loop
string name_for_loop(Stmt s) {
  ContainForLoop cfl;
  s.accept(&cfl);
  return cfl.varnames[0];
}

// Identifies all for loop names in code statemen
vector<string> contained_for_loop_names(Stmt s) {
  ContainForLoop cfl;
  s.accept(&cfl);
  return cfl.varnames;
}


class UsesVariable : public IRVisitor {
  using IRVisitor::visit;
  void visit(const Variable *op) {
    if (op->name == varname) {
      used = true;
    }
    return;
  }

  void visit(const Call *op) {
    // only go first two variables, not loop bour
    if (op->name == "write_stream" && op->args.si
      op->args[0].accept(this);
      op->args[1].accept(this);
    } else {
      IRVisitor::visit(op);
    }
  }

public:
  bool used;
  string varname;
  UsesVariable(string varname) : used(false), var
};

// identifies target variable string in code stat
bool variable_used(Stmt s, string varname) {
  UsesVariable uv(varname);
```



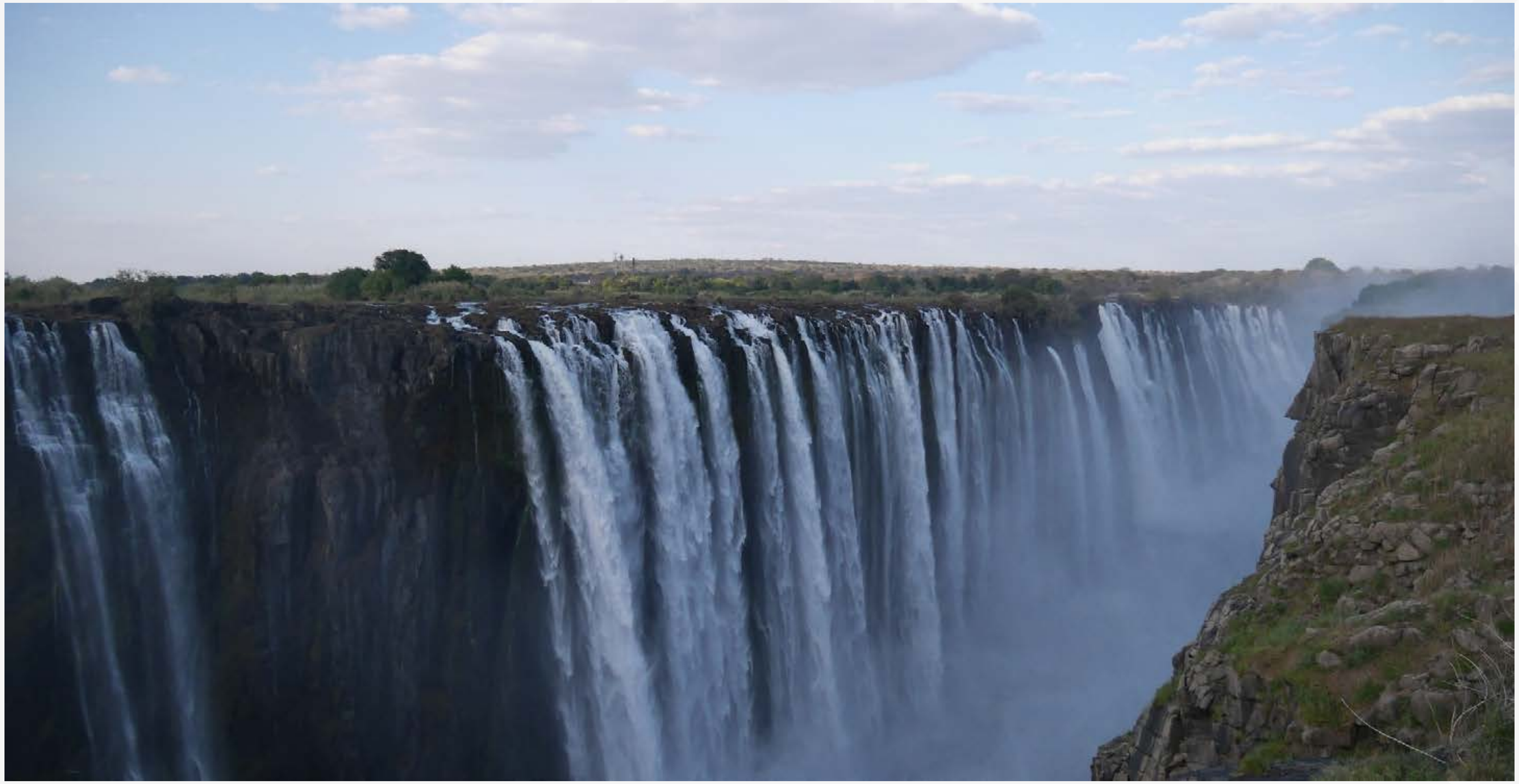
```
class Demosaic : public Halide::Generator<Demosaic> {
public:
    GeneratorParam<LoopLevel> intermed_compute_at{"intermed_compute_at", LoopLevel::inlined()};
    GeneratorParam<LoopLevel> intermed_store_at{"intermed_store_at", LoopLevel::inlined()};
    GeneratorParam<LoopLevel> output_compute_at{"output_compute_at", LoopLevel::inlined()};

    // Inputs and outputs
    Input<Func> deinterleaved{ "deinterleaved", Int(16), 3 };
    Output<Func> output{ "output", Int(16), 3 };

    // Defines outputs using inputs
    void generate() {
        // These are the values we already know from the input
        // x_y = the value of channel x at a site in the input of channel y
        // gb refers to green sites in the blue rows
        // gr refers to green sites in the red rows

        // Give more convenient names to the four channels we know
        Func r_r, g_gr, g_gb, b_b;

        g_gr(x, y) = deinterleaved(x, y, 0);
        r_r(x, y)  = deinterleaved(x, y, 1);
        b_b(x, y)  = deinterleaved(x, y, 2);
        g_gb(x, y) = deinterleaved(x, y, 3);

        // These are the ones we need to interpolate
        Func b_r, g_r, b_gr, r_gr, b_gb, r_gb, r_b, g_b;

        // First calculate green at the red and blue sites

        // Try interpolating vertically and horizontally. Also compute
        // differences vertically and horizontally. Use interpolation in
        // whichever direction had the smallest difference.
        Expr gv_r  = avg(g_gb(x, y-1), g_gb(x, y));
        Expr gvd_r = absd(g_gb(x, y-1), g_gb(x, y));
        Expr gh_r  = avg(g_gr(x+1, y), g_gr(x, y));
        Expr ghd_r = absd(g_gr(x+1, y), g_gr(x, y));

        g_r(x, y)  = select(ghd_r < gvd_r, gh_r, gv_r);

        Expr gv_b  = avg(g_gr(x, y+1), g_gr(x, y));
```

NOT SO SECRET DOWNSIDE

$100M

Many Years

NOT SURPRISING



It is a waterfall model of design!

SOFTWARE ISN'T BUILT THAT WAY

- Moved away from that style decades ago
- Enables small teams to build amazing apps

AGILE DESIGN

Rapidly iterate on end-to-end system

Learn about real problems, and goals

AHA!

Agile Hardware

AGILE DESIGN

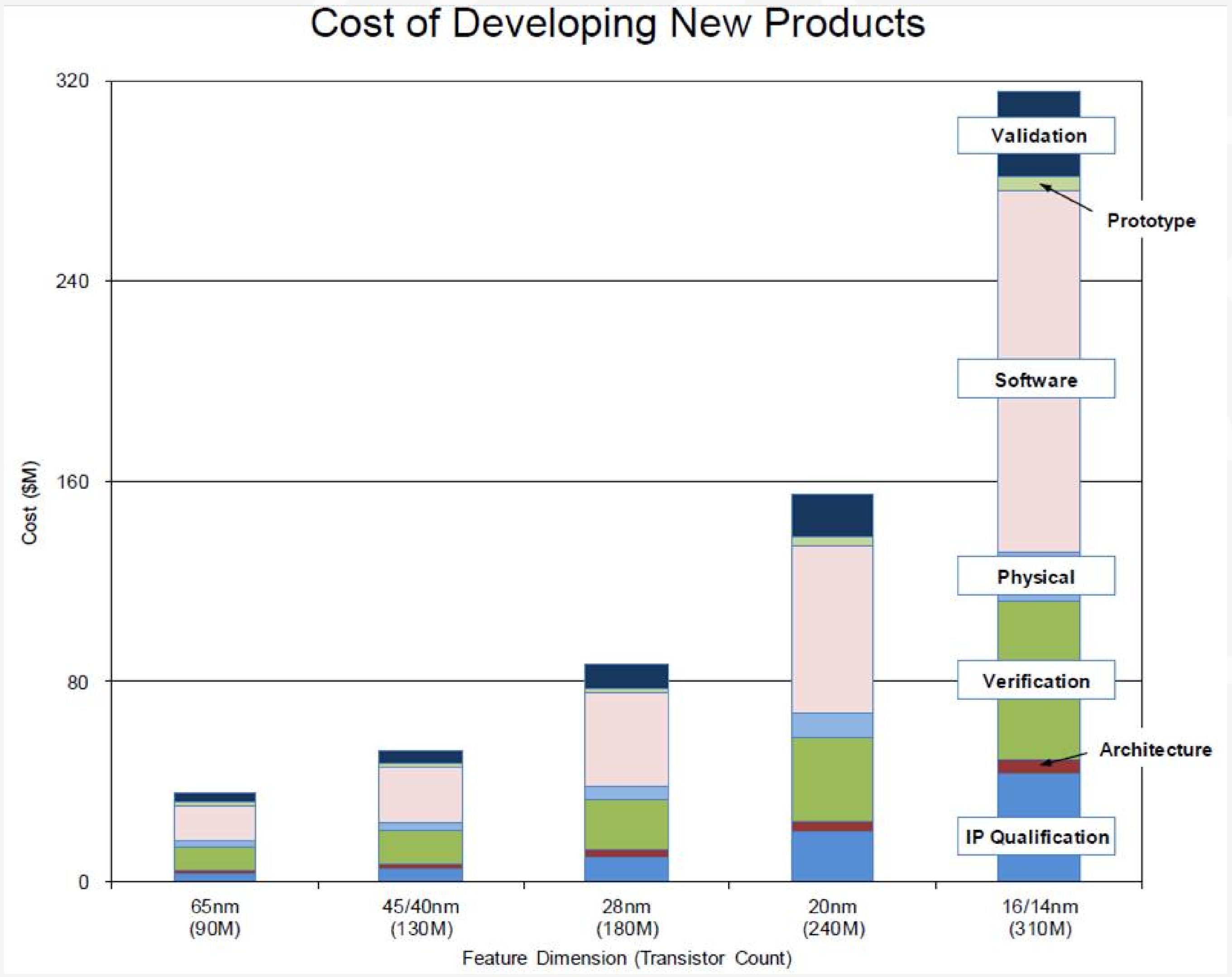
It is about reuse

It is about clean interfaces

It is about constructors, not instances

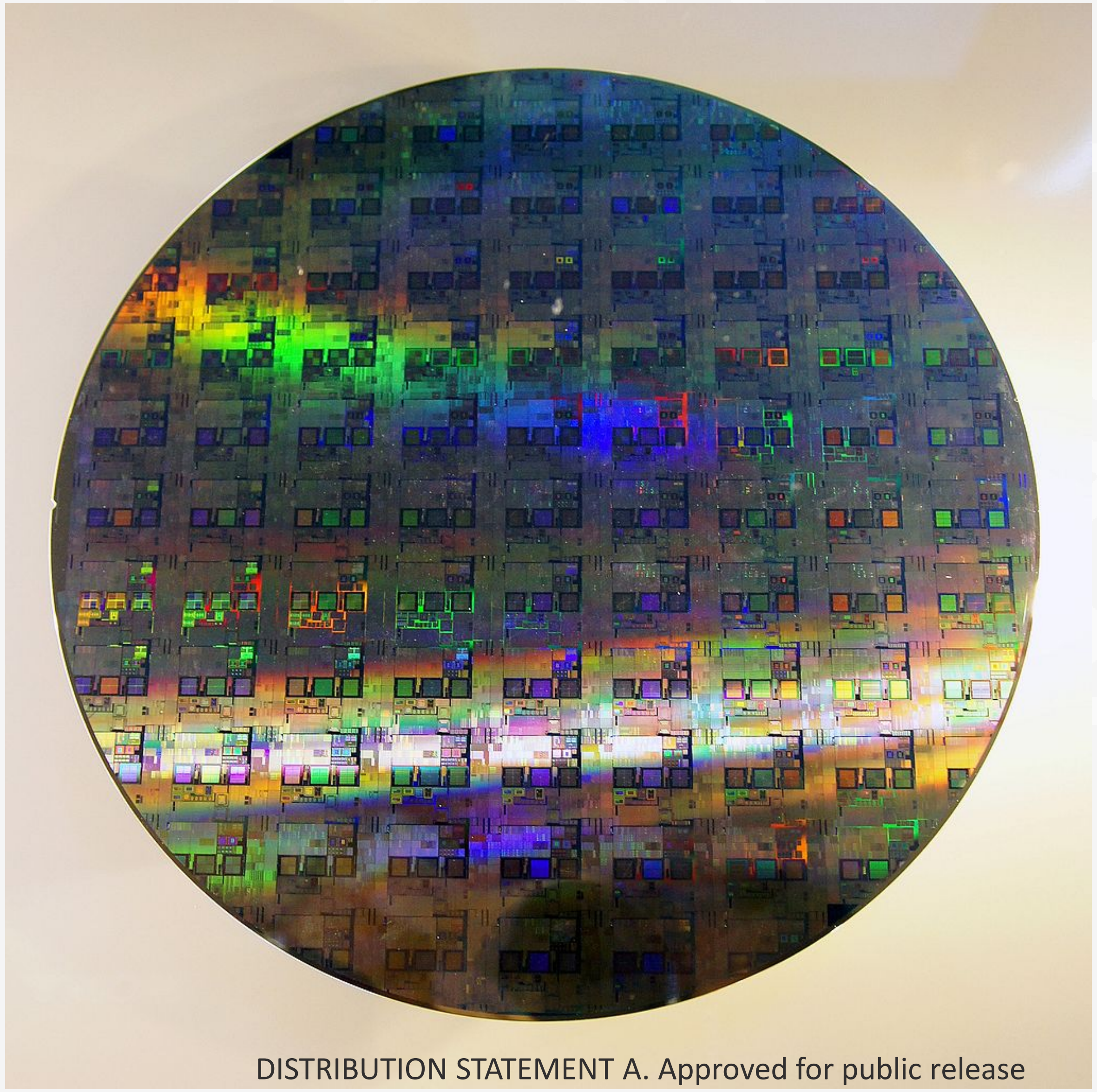
It is about productivity

PRODUCTIVITY IS THE ISSUE IN HARDWARE



Source: IBS

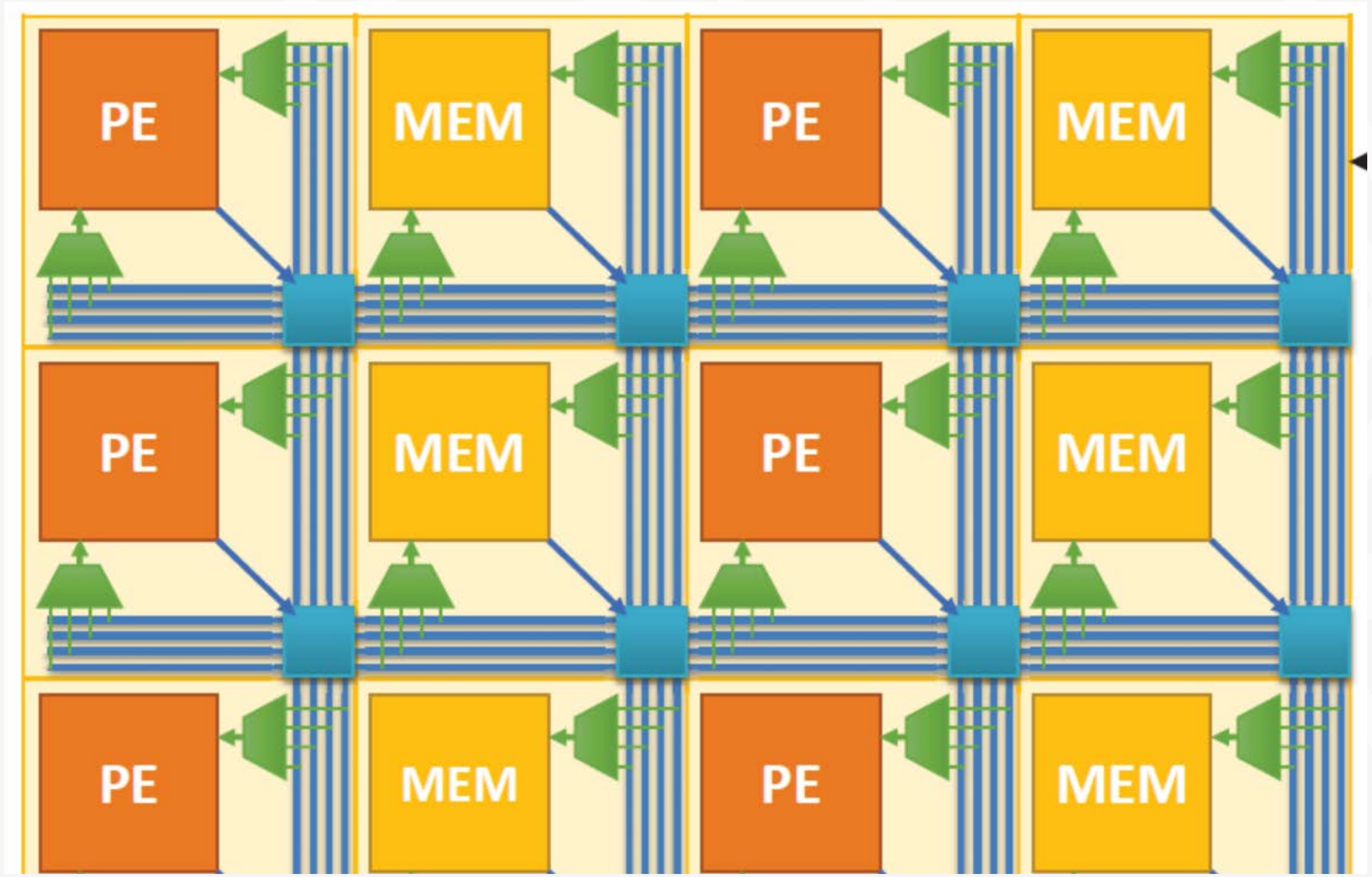
STILL NEED TO DEAL WITH FABRICATION



https://upload.wikimedia.org/wikipedia/commons/thumb/e/eb/12-inch_silicon_wafer.jpg/1024px-12-inch_silicon_wafer.jpg

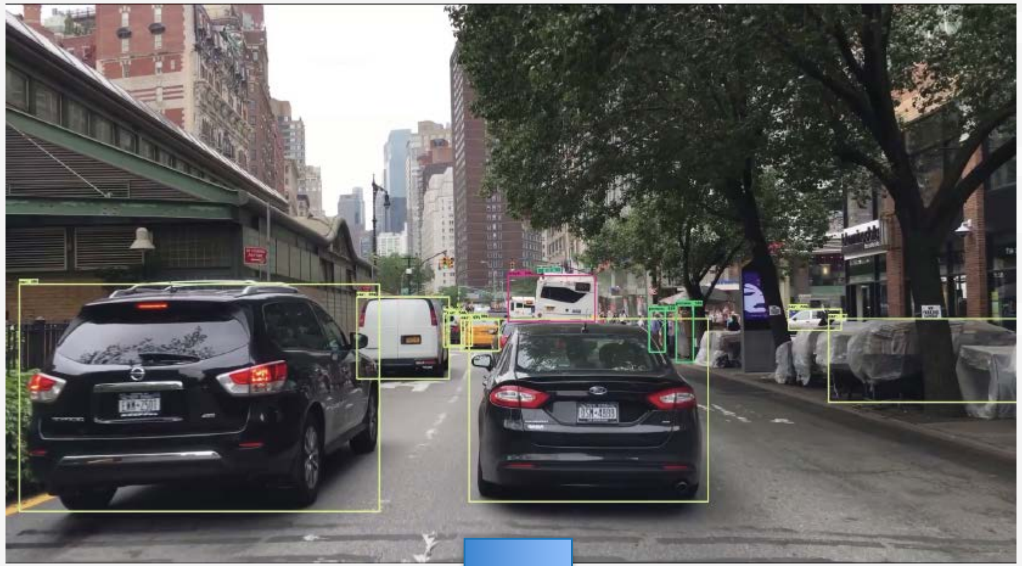
NEED TO EVOLVE THE HARDWARE

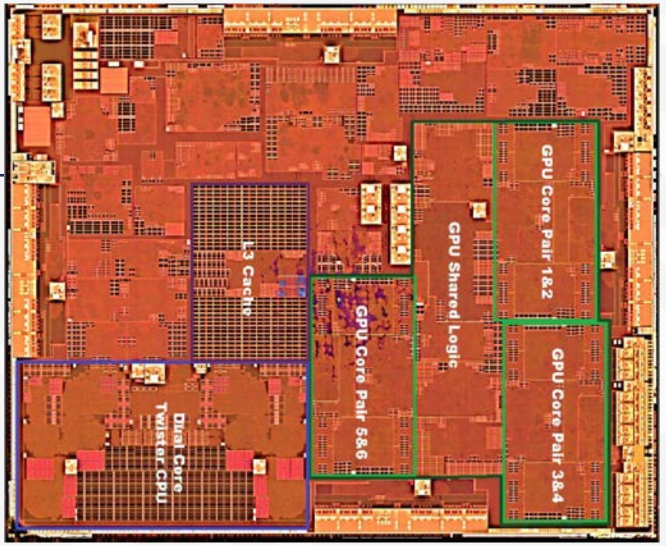
- Use a CGRA – a configurable framework



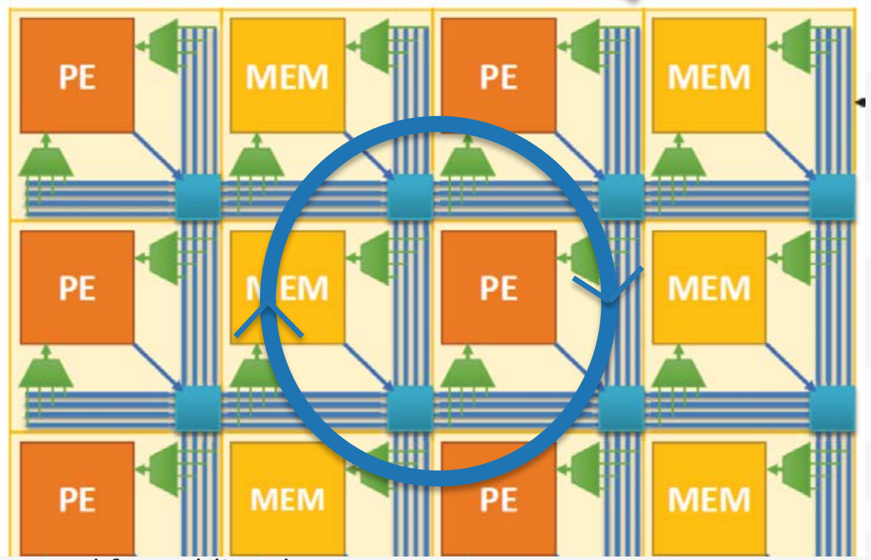
AHA VISUAL COMPUTING

A new way to create DSSoCs





Evolves

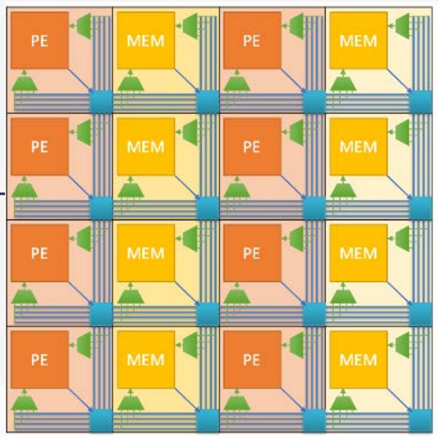


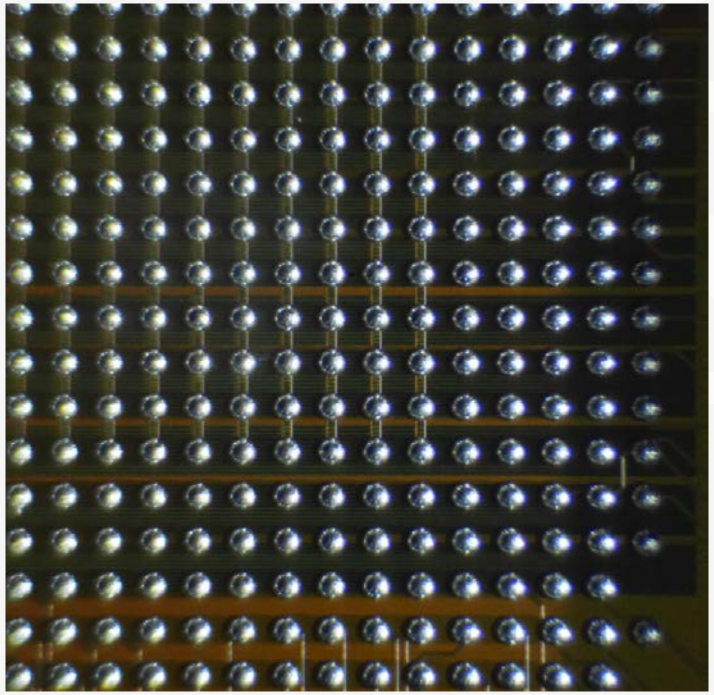
Optimize

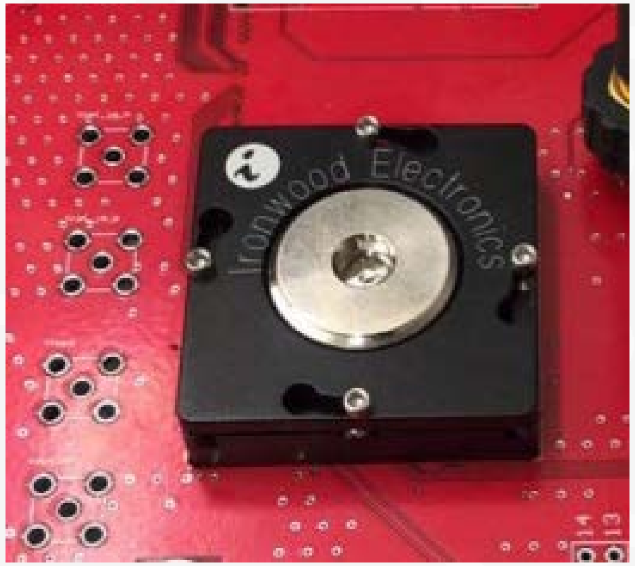
Compile

DISTRIBUTION STATEMENT A. Approved for public release

FIRST GENERATION CGRA

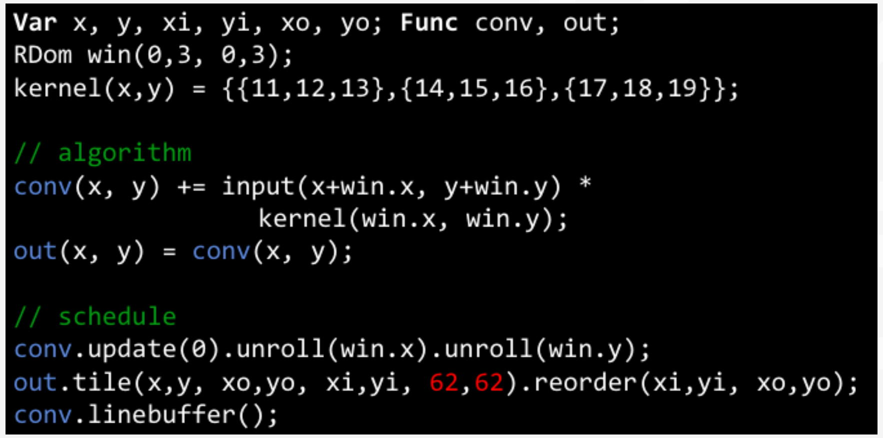






APPLICATION COMPILATION FLOW

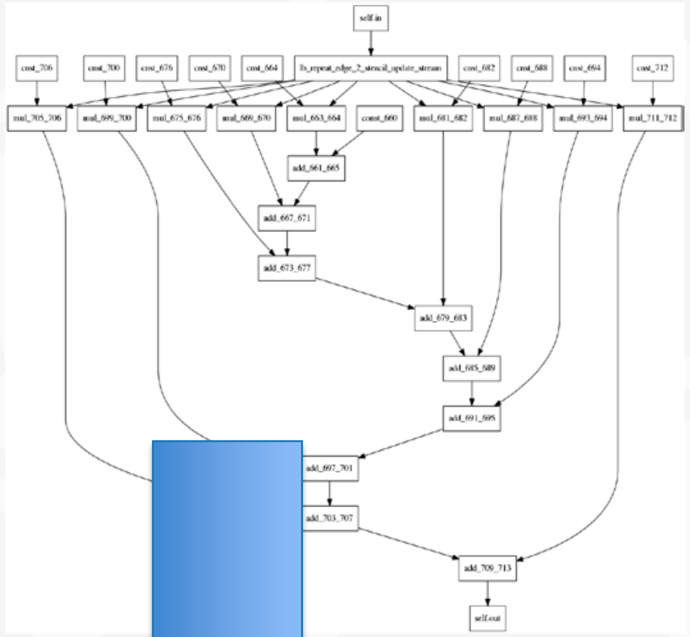
Halide



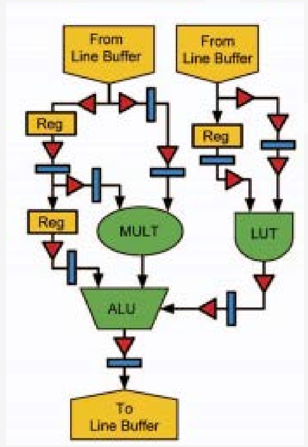
```
Var x, y, xi, yi, xo, yo; Func conv, out;
RDom win(0,3, 0,3);
kernel(x,y) = {{11,12,13},{14,15,16},{17,18,19}};

// algorithm
conv(x, y) += input(x+win.x, y+win.y) *
                 kernel(win.x, win.y);
out(x, y) = conv(x, y);

// schedule
conv.update(0).unroll(win.x).unroll(win.y);
out.tile(x,y, xo,yo, xi,yi, 62,62).reorder(xi,yi, xo,yo);
conv.linebuffer();
```

Compiler

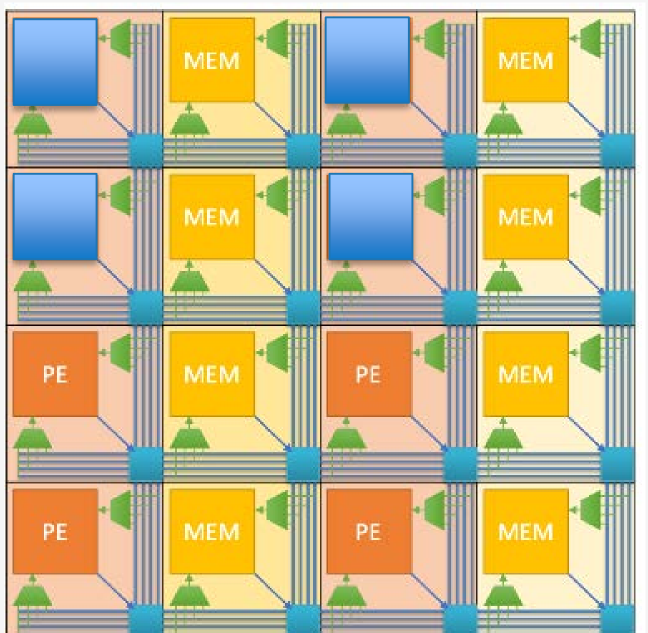
CoreIR



Mapping



Place & Route



MAINTAINING THE FLOW THROUGH CHANGES

- Many tools need to know about your design
  - You are building a “world”

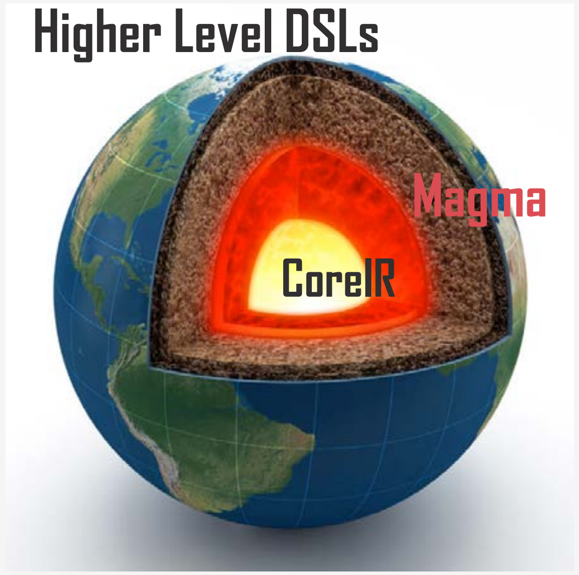


Image source: Jesse Varner - Raft Lake and the Little Wind River Valley beyond, https://www.flickr.com/photos/molas/1364382020/in/photostream/

DSSOC DOESN’T NEED TO BE EXPENSIVE

- One just needs to think about the problem differently
- We have already created one working chip/system using this flow
- And have the next generation of the system working

Stay Tuned for Future Results …

ERI ELECTRONICS RESURGENCE INITIATIVE SUMMIT

2019 | Detroit, MI | **July 15 - 17**