Tensilica Day

A Flexible ASIP Architecture for Connected Components Labeling: Implementation, Lessons Learned, and Integration into Novel Design Tools

Juan Fernando Eusse and Rainer Leupers
Hannover, February 9th 2016

ICE | RWTH AACHEN UNIVERSITY

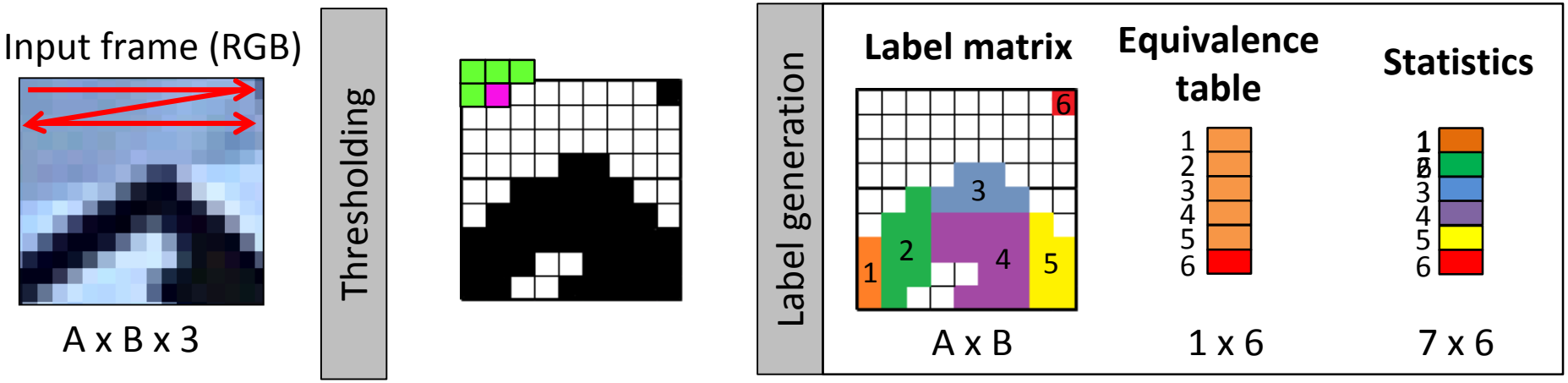
Agenda

1. Connected Components Labeling ASIP
2. Methodological Observations
3. Pre-Architectural Performance Estimation
4. Conclusions

Connected Components Labeling (CCL)

- Detect connected regions of pixels
  - Single pass algorithm
    - One iteration over the input frame
    - Additional data structures (memory) required
  - Rasterized processing of individual pixels
  - Uses 8 nearest neighbor mask
  - Collect region characteristics on-the-fly (merge at the end)

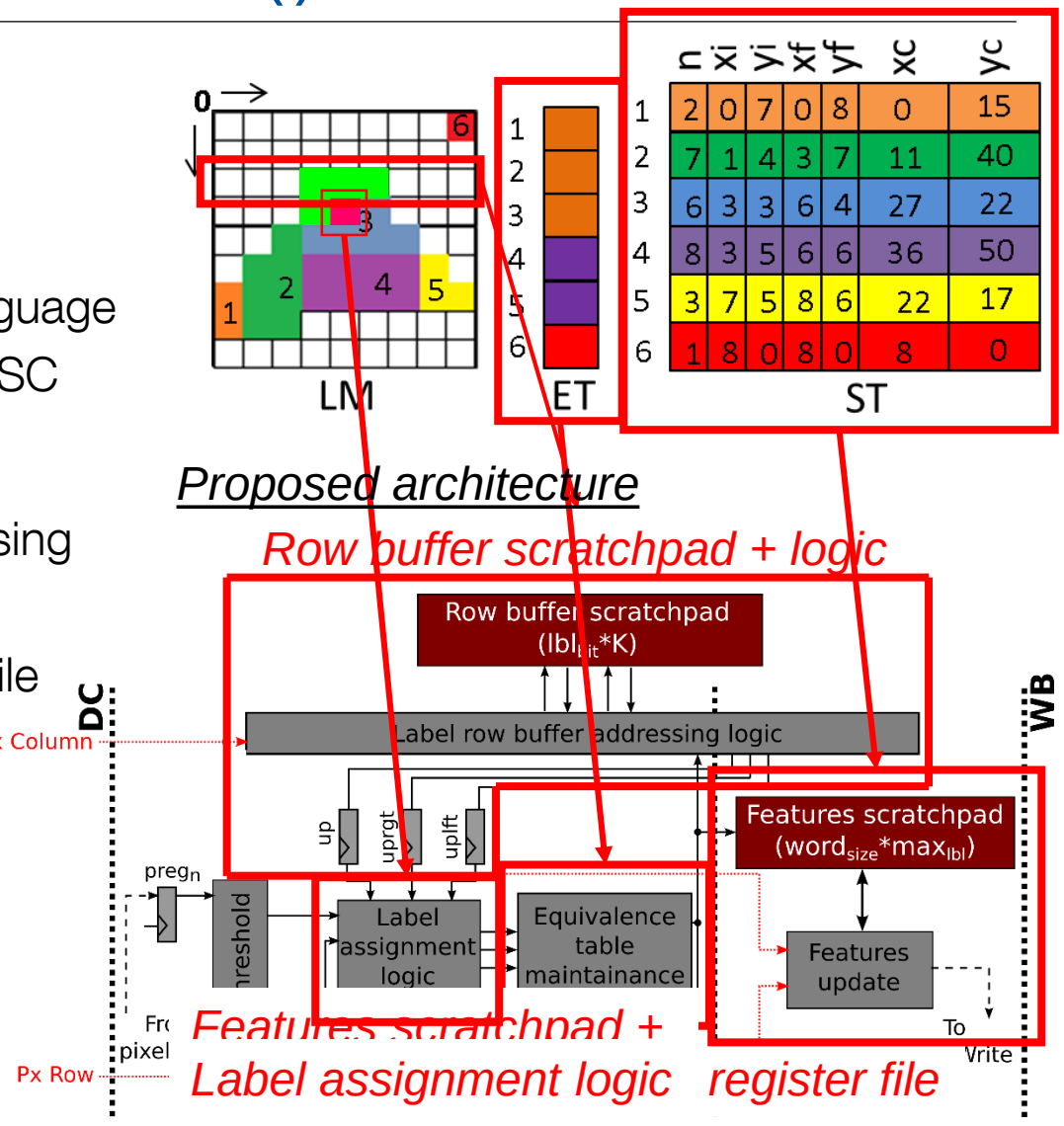
RICOH



ICE | RWTH AACHEN UNIVERSITY

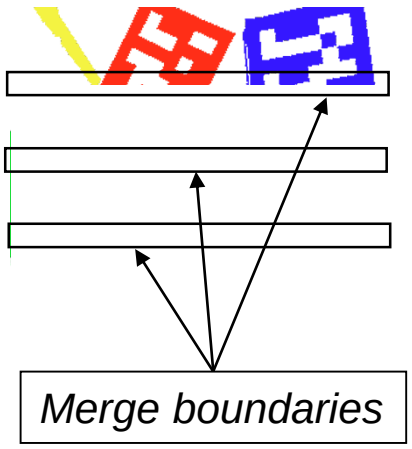
Architectural Customizations (I)

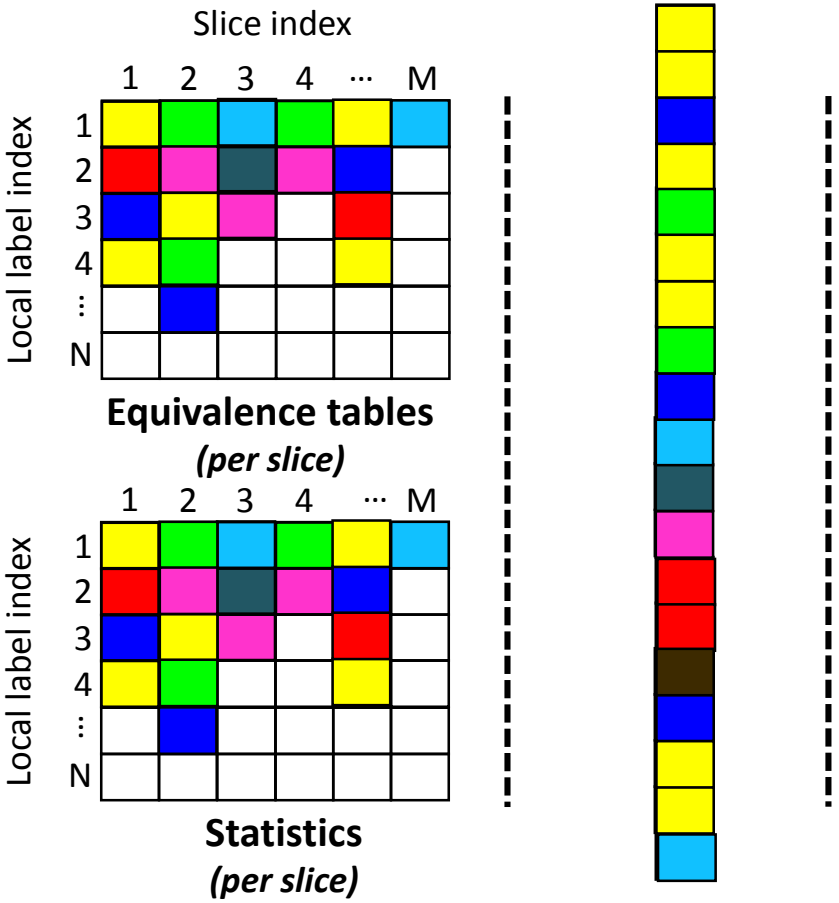
- Customization of a template architecture
  - Iterative architecture exploration
  - LISA architecture description language
  - Synopsys Processor Designer RISC
- Added custom logic
  - Row buffer scratchpad + addressing
  - Label assignment logic
  - ET maintenance logic + register file
  - Features scratchpad + update logic
- HW size dependent on
  - Frame size
  - Number of possible labels



Algorithmic Modification: Slicing Approach

- Hardware size explodes with frame size/complexity





N: Max. labels per slice
M: Slices per frame

num	xi	yi	xf	yf	xc	yc
2	0	7	0	8	0	15
7	1	4	2	7	11	40
6	3	3	6	4	27	22
8	3	5	6	6	36	50
3	7	5	8	6	22	17
1	8	0	8	0	8	0

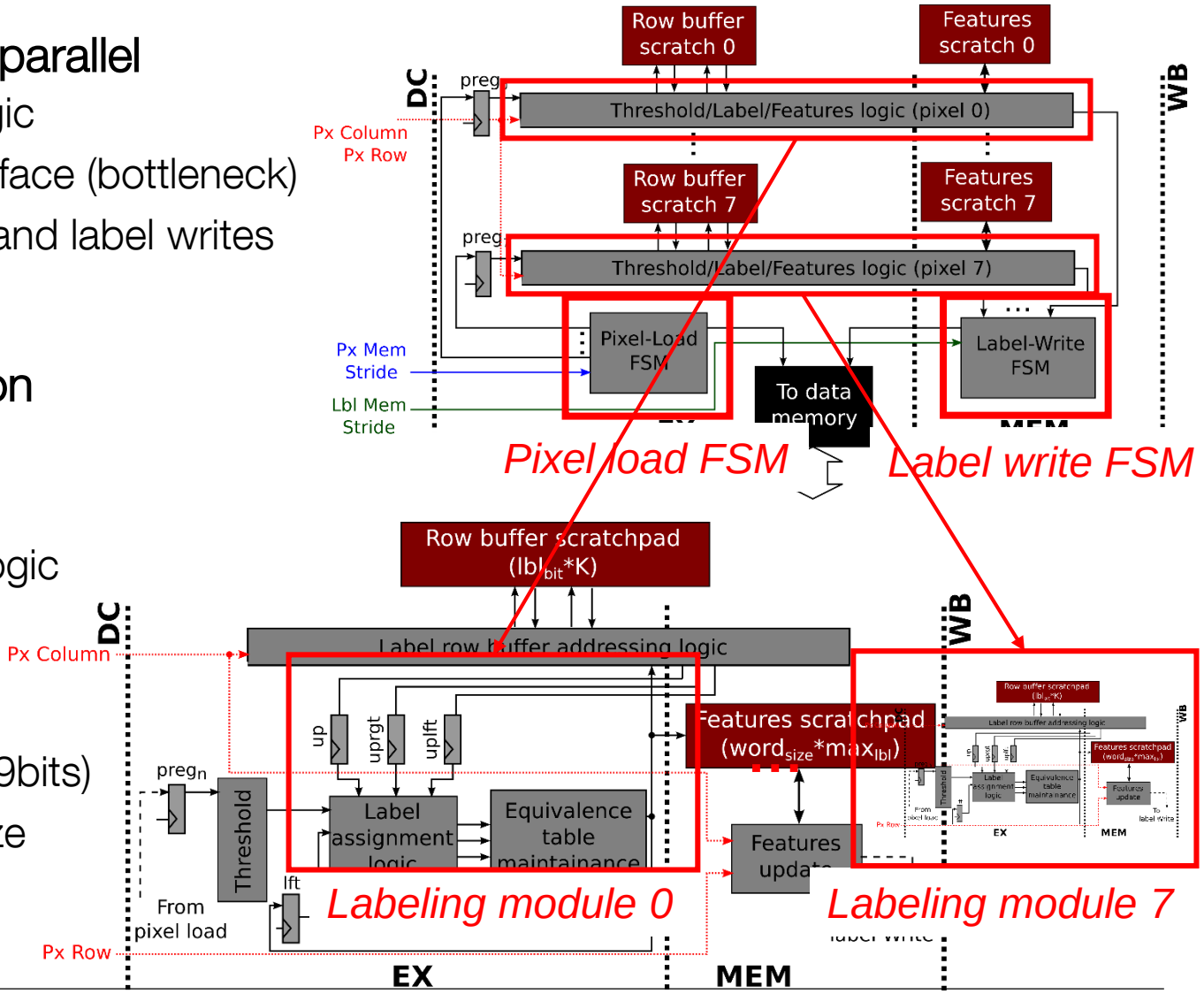
1. Process slices

2. Merge slice labels

3. Calculate final statistics

Architectural Customizations (II)

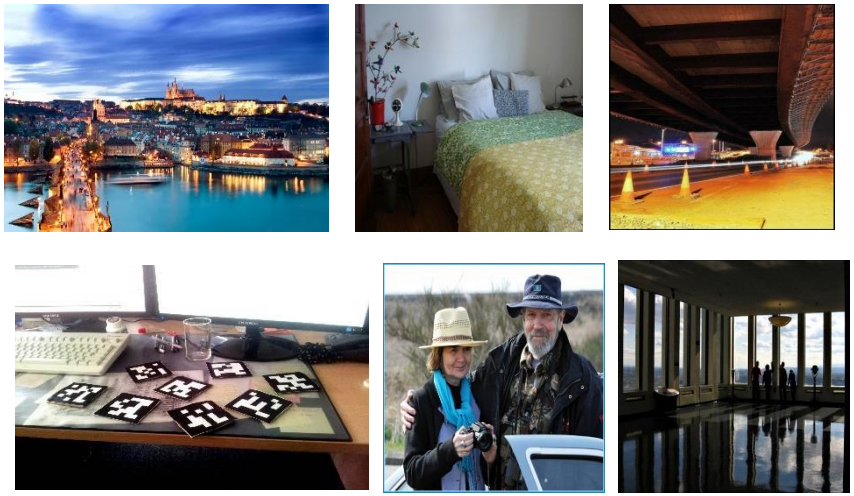
- **Process M pixels in parallel**
  - Replicate custom logic
  - Standard 32-bit interface (bottleneck)
  - Serialize pixel reads and label writes
- **Further customization**
  - Pixel load FSM
  - Label write FSM
  - 8x custom labeling logic
- **Chosen parameters**
  - 512 labels per slice (9bits)
  - 2048x2048 frame size



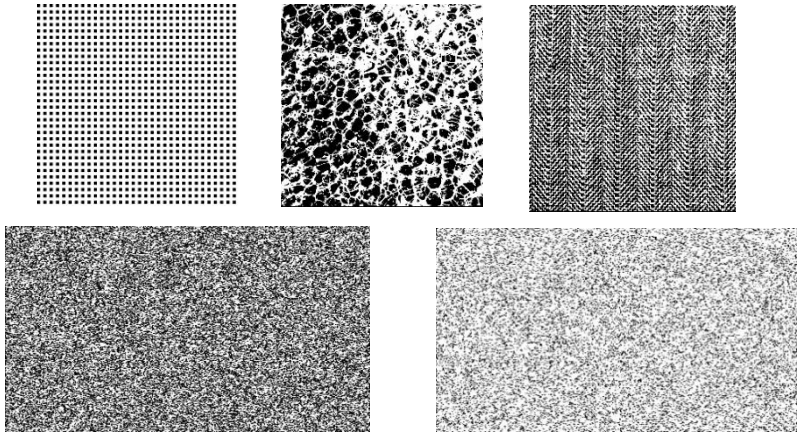
Experimental Results: Setup and Metric Definition

- **Input data sets**
  - CCL performance influenced by frame complexity
  - Publicly available frame sets
  - Synthetic and natural images
- **Performance metric**
  - *Cycles-per-pixel (cpp)* characterize architectural efficiency
  - Independent from core frequency
- **Simulation setup**
  - Cycle accurate simulator used
  - *Best/average/worst* ***cpp*** obtained

Natural images

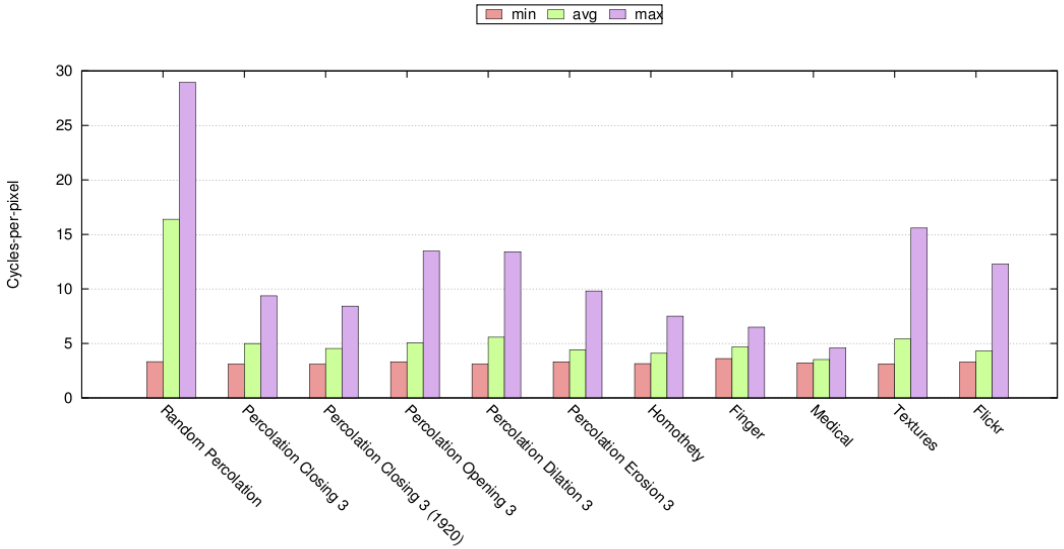


Synthetic data sets

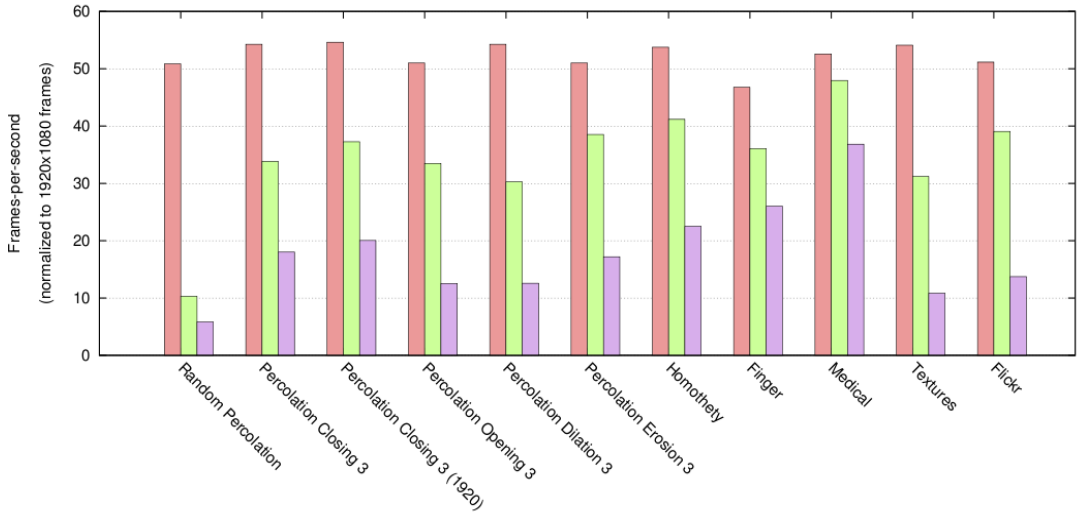


Experimental Results: Synthesis and Performance

- Synthesized with design compiler
  - *350MHz@65nm* – 1.8*mm²*
  - Estimated power consumption
    - 110mW@25°C
    - 228mW@125°C



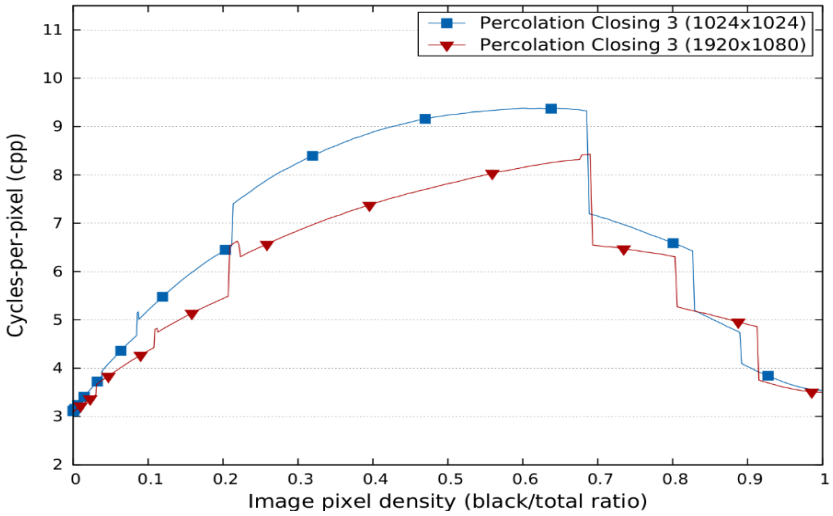
- Both *cpp* and *fps* metrics used
  - >30*fps* in avg for practical images
  - >10*fps* worst case for most synthetic data sets
  - 5*fps* in the absolute worst case



ICE | RWTH AACHEN UNIVERSITY

Experimental Results: Flexibility and Comparison

- Impact of size and frame complexity
  - cpp variation given frame complexity
  - Super-slicing via SW (penalty observed)
- Performance comparison against
  - Original PD_RISC (base)
  - TI TMS320C64x DSP
- Performance gains (Flickr)
  - PD_RISC: 6.7/33.1/87.8
  - TI DSP: 10.2/11.6/12.9



		Ours	PD_RISC Base (cpp)	Gain	TMS320C64x (cpp)	Gain
Homothety	*min*	3.14	22.0	7.01	39.76	12.66
	avg	4.11	26.28	6.39	42.70	10.38
	max	7.54	40.88	5.42	50.60	**6.71**
Random Percolation	*min*	3.31	17.16	**5.18**	25.59	7.73
	avg	16.39	3,524	215	394.42	24.06
	max	28.9	10,384	**359**	1,107.53	**38.32**
Finger	*min*	3.61	21.8	6.04	35.81	9.2
	avg	5.03	84.07	16.71	53.79	10.69
	max	6.49	167.48	25.8	64.87	10.0
Textures	*min*	3.12	19.75	6.33	30.86	9.89
	avg	7.5	493.69	65.82	106.12	14.15
	max	15.59	2,219.36	142.35	303.20	19.45
Flickr	*min*	3.3	22.38	6.78	33.71	10.21
	avg	6.49	215.3	33.17	75.25	11.60
	max	12.29	1,080.14	87.88	158.56	12.9

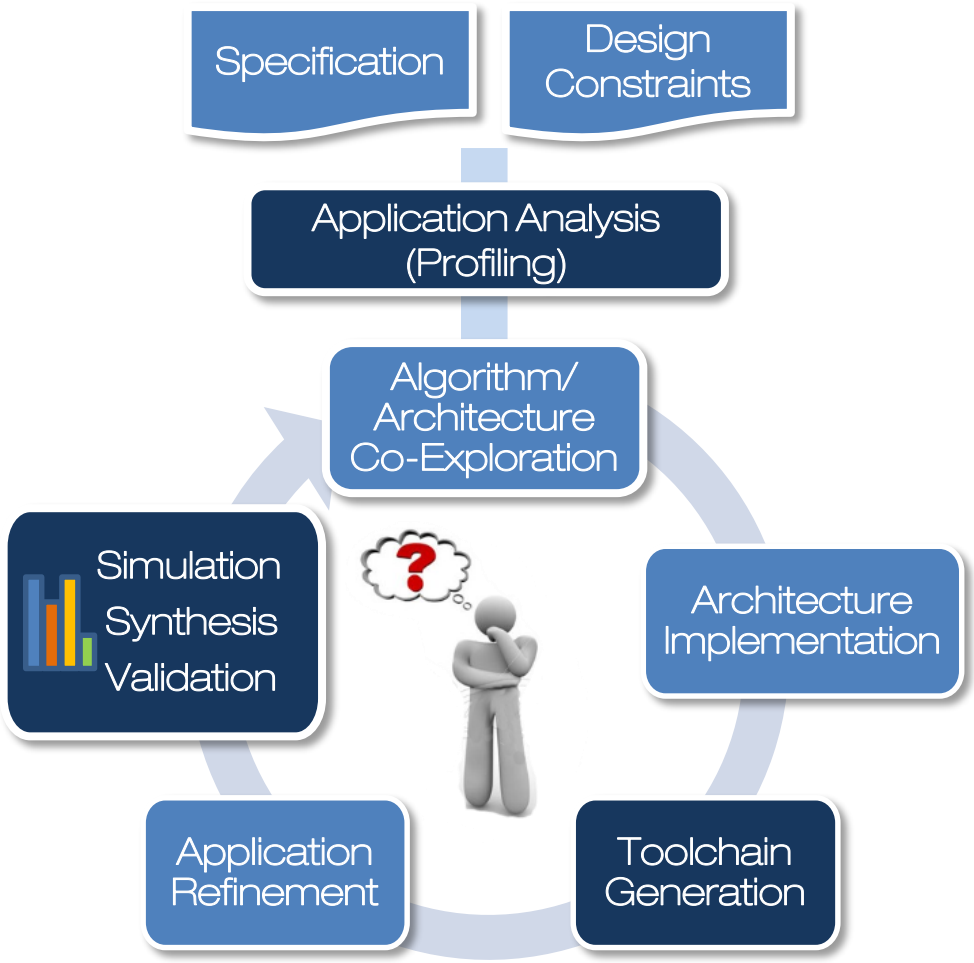
Agenda

1. Connected Components Labeling ASIP
2. Methodological Observations
3. Pre-Architectural Performance Estimation
4. Conclusions

Methodological Observations: Design Gap

- Current design methodologies
  - Semi-automate some design steps
  - ASIP performance obtained through simulation
    - Last step of the design cycle
  - No warranty of requirement satisfaction

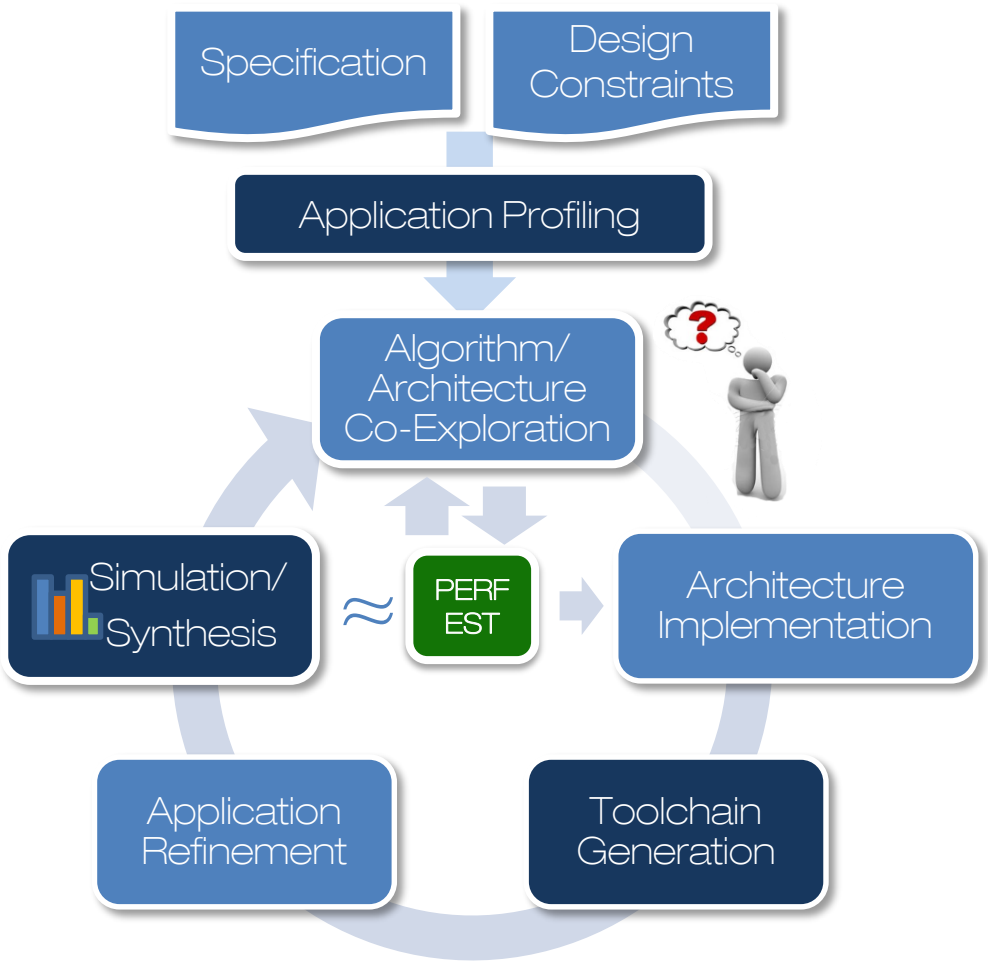
Several design iterations required!!!



- There is a need for tools that:
  - Increase the chance of *first-time-architectural* success

Methodological Observations: Bridging the Gap

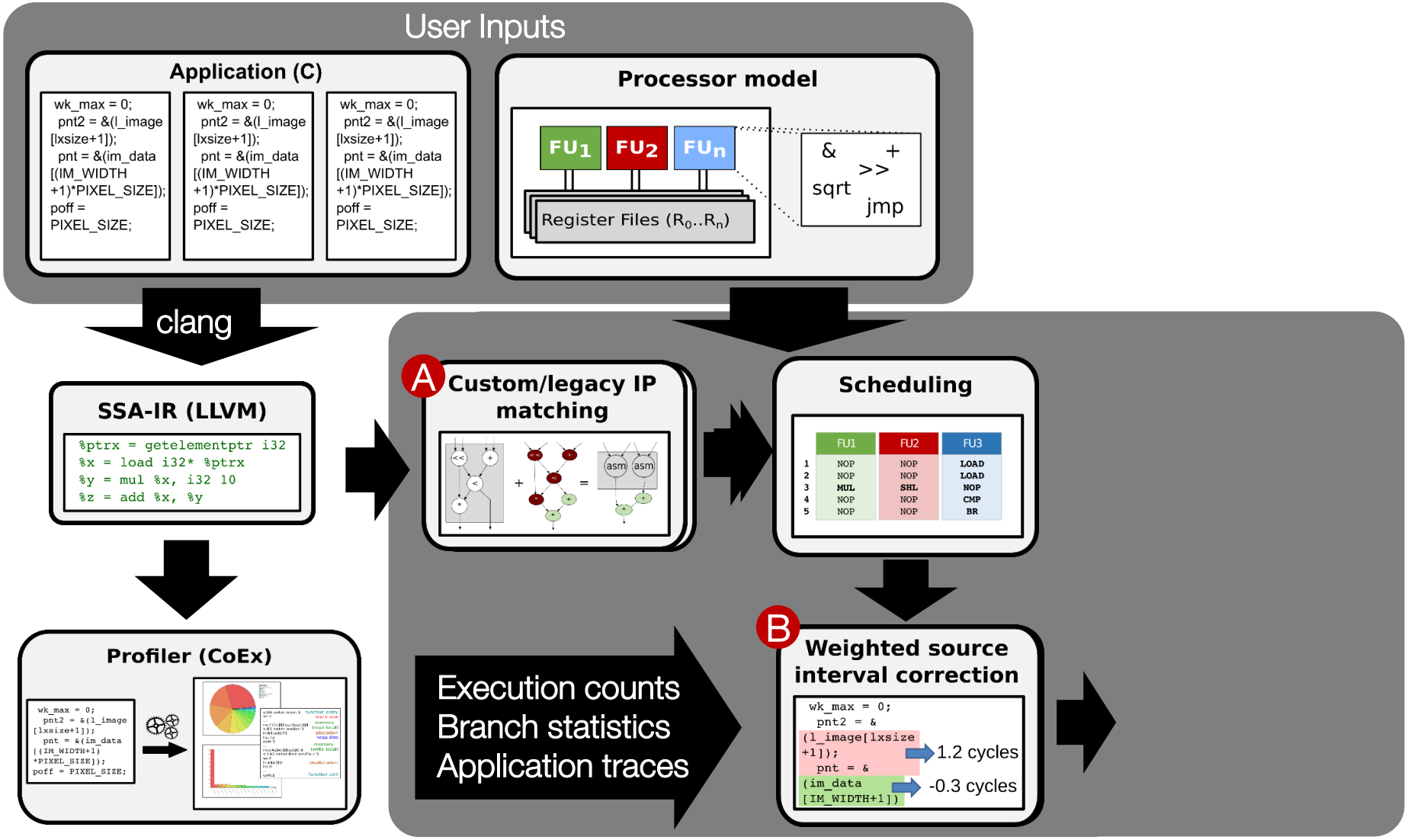
- **Pre-architectural estimation of achievable performance**
  - Use high level models to predict application cycles
  - Reduce the number of complete design iterations
  - Complement existing design flows



Agenda

1. Connected Components Labeling ASIP
2. Methodological Observations
3. Pre-Architectural Performance Estimation
4. Conclusions

Performance Estimation: Datapath



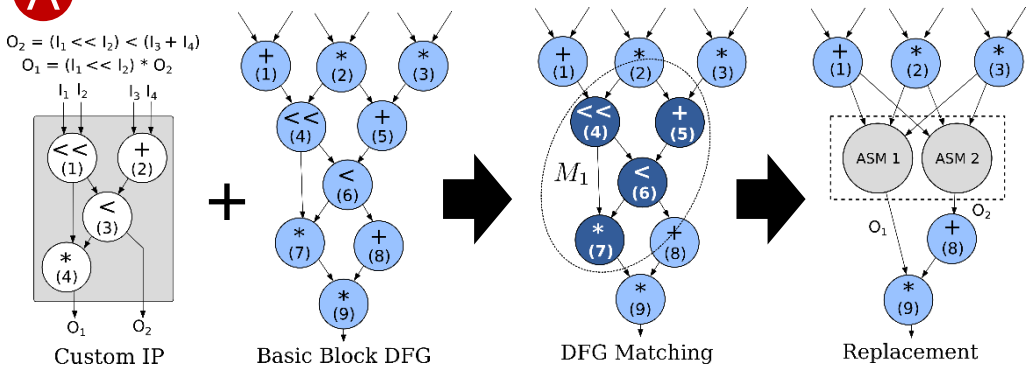
Performance Estimation: Datapath (II)

- Base estimation only covers:
  - Architecture selection
  - Instruction set design

- Does a HW modification improve performance?
  - Discard sub-optimal mods
  - Analyze side effects

- Customization techniques to support:
  - A Custom instructions/Legacy IP
  - B What-if scenarios based on code intervals

A



B

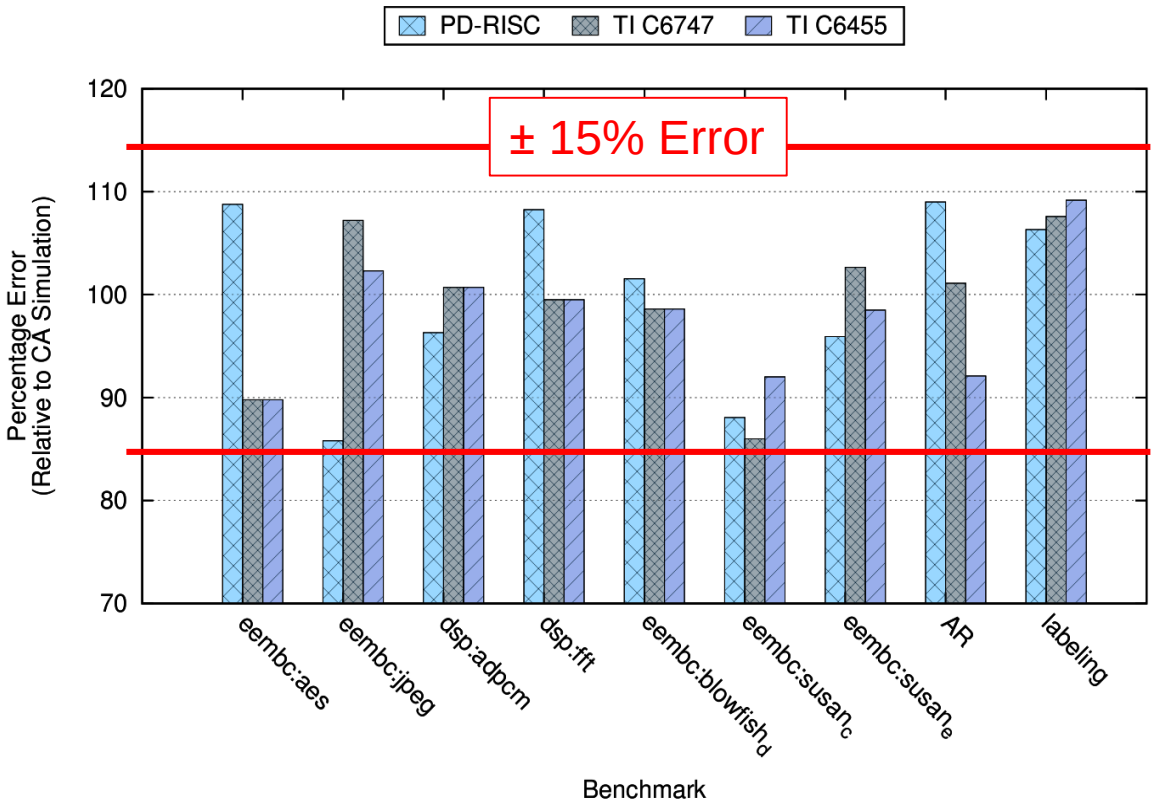
```
if(isBlack)
{
  pnt1 = &(pnt3[-IM_WIDTH]);
  if( *pnt1 > 0 )
  {
    // Update the temporal matrix and the label matrix
    *pnt3 = *pnt1;
    *pnt2 = *pnt1;

    int point_index = ((*pnt3)-1)*7;

    ST_slice_ptr[point_index+0] ++;
    ST_slice_ptr[point_index+1] += i;
    ST_slice_ptr[point_index+2] += j;
    ST_slice_ptr[point_index+6]  = j;
  }
  else if( *(pnt1+1) > 0 ) {
    if( *(pnt1-1) > 0 ) {
      m = CC_slice_ptr[*(pnt1+1)-1];
      n = CC_slice_ptr[*(pnt1-1)-1];
      if( m > n ) {
        *pnt3 = n;
```

Cost = 1 cycle (Custom Memories)

Cost = Est*0.5 **cycles (2x Parallelism)**

Performance Estimation: Accuracy

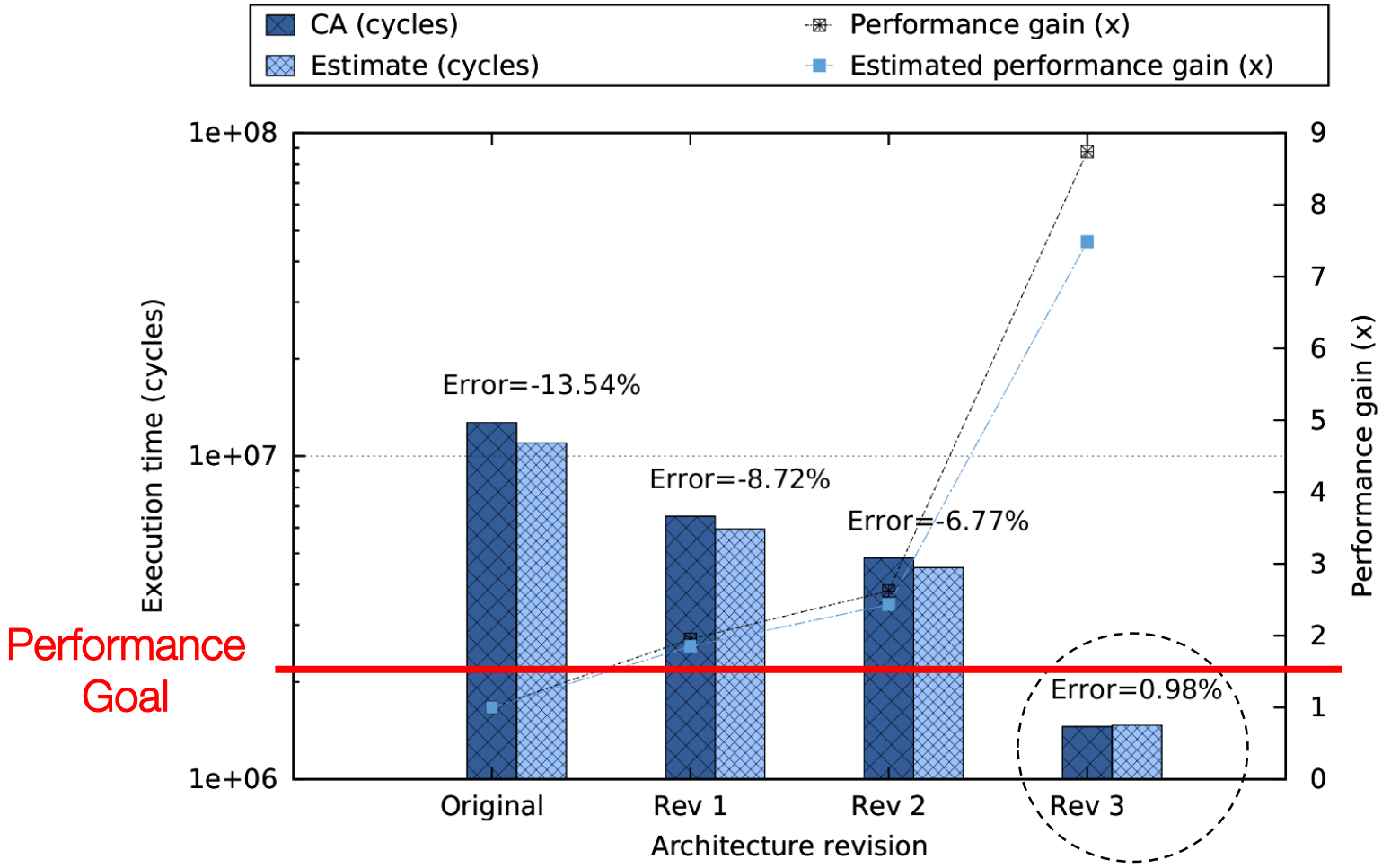
- Usability depends on estimation accuracy
- Several commercial processors
  - PD_RISC (Synopsys)
  - C67x/C64x/C66x (TI DSPs)
- Using Cycle Accurate Simulators
  - Flat memory model, no caching
- Integrated by *Silexica* as a general purpose estimator
  - ARM A7/A9/A15/M4 and Adapteva's Epiphany models
  - Parallel application mapping into heterogeneous MPSoCs



Average gain: 248x (PD-RISC), **67x** (TI DSPs)
(CA sim. time Vs. profiling + estimation time)

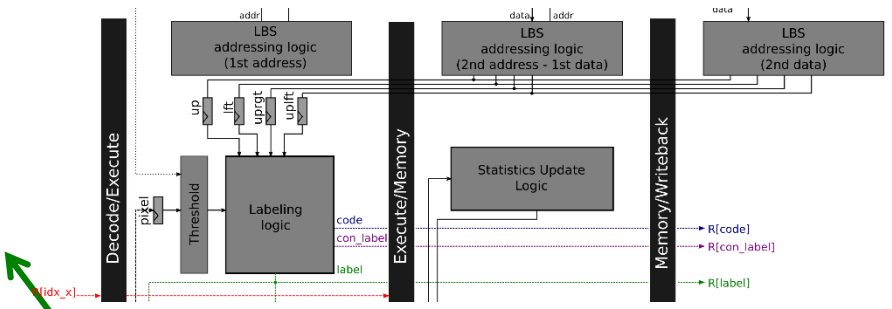
ICE | RWTH AACHEN UNIVERSITY

Performance Estimation: ASIP Design



Performance Estimation: ASIP Design (I)

```
1  for sl = 0 . . . Slices do
2    for y = 0 . . . height/Slices do
3        for x = 0 . . . width do
4            pixel = (x, sl.Slices + y)
5            black = Image(pixel) < Threshold
6            if not black then
7              label(i, j) ← 0
8            else
9              if label(N) > 0 then
10             label(pixel) ← label(N)
11             else if label(NE) > 0 then
12                 if label(NW) > 0 then
13                     mergedLabel ← merge(label(NW,NE))
14                     label(pixel) ← mergedLabel
15                 else if label(W) > 0 then
16                     mergedLabel ← merge(label(NW,W))
17                     label(pixel) ← mergedLabel
18                 else
19                     label(pixel) ← label(NE)
20                 end if
21             else if label(NW) > 0 then
22                 label(pixel) ← label(NW)
23             else if label(W) > 0 then
24                 label(pixel) ← label(W)
25             else
26                 create new label
27             end if
28           end if
29       end for
30   end for
31 end for
```



Custom Instruction Interval (Fixed Cost = 9)

Exclusion zone
→ Use original costs from estimates

```
<SourceWeighting name="loop" function="main" type="fixed" cost="9">
    <SourceRegion begin="4" end="28" />
    <Exclusion begin="11" end="17" />
    <Exclusion begin="25" end="28" />
</SourceWeighting>
```


ICE | RWTH AACHEN UNIVERSITY

Agenda

1. Connected Components Labeling ASIP
2. Methodological Observations
3. Pre-Architectural Performance Estimation
4. Conclusions

Conclusions

- **Created an ASIP capable to perform CCL:**
  - Solution supports arbitrary frame sizes with varying complexity
  - Capable of labeling FullHD frames at 45/30/5 *fps* in the *best/average/worst* case
  - Evaluation performed over an extensive data set of over 11000 images
  - Outperforms a commercial TI DSP for a factor of *10x*
- **Based on the performed ASIP design:**
  - Realized a set of tools that enable high level performance estimation based on abstract processor models
  - Obtained accuracies up to ±*15%* for the modeled processors
  - Estimation is up to *248x* faster than cycle accurate simulation
  - Currently being applied by **Silexica Software Solutions GmbH**
    - Estimation used for MPSoC task mapping decisions
    - New processor models being created (ARM, Epiphany)

Thank you!

Questions?