Exploiting Dynamic Timing Slack for Energy Efficiency in Ultra-low-power Embedded Systems

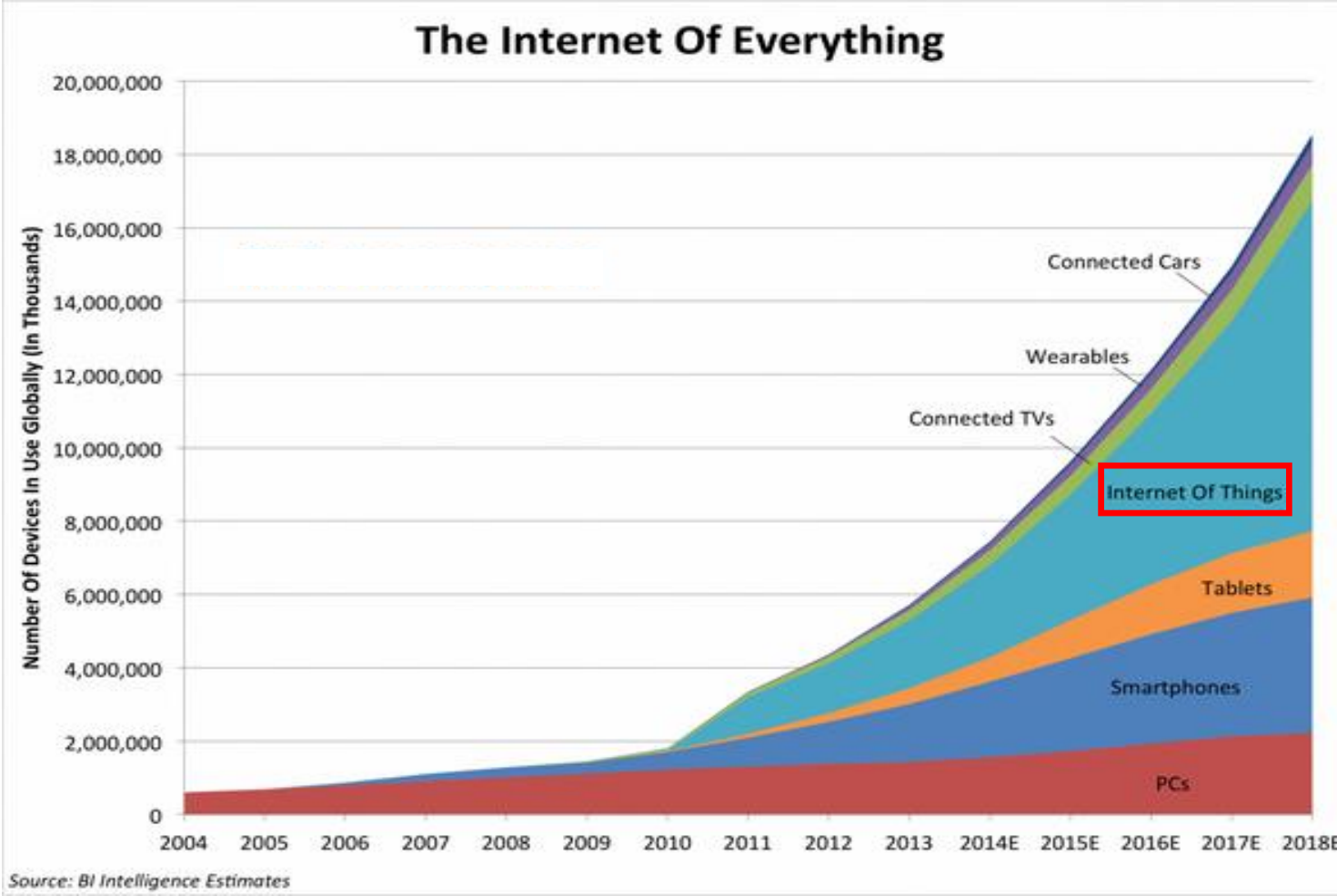
Hari Cherupalli - *Minnesota*

Rakesh Kumar - *Illinois*

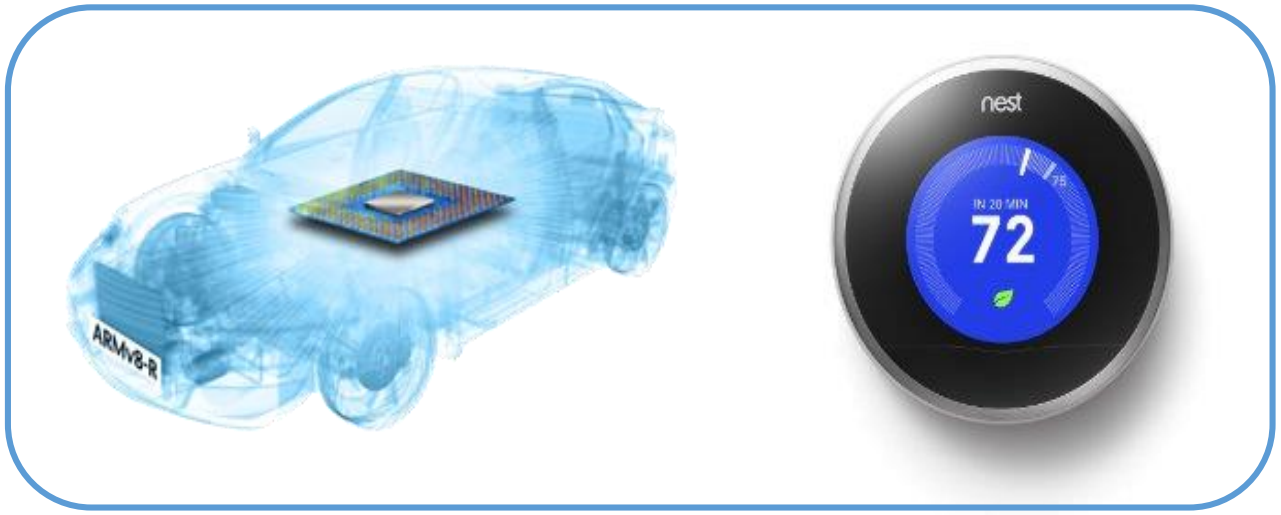
John Sartori - *Minnesota*

Why Ultra-low-power Embedded Systems?

- Low-power µPs and µCs are the **most abundant** type of processors
- Saving power for these processors can have a **huge impact**



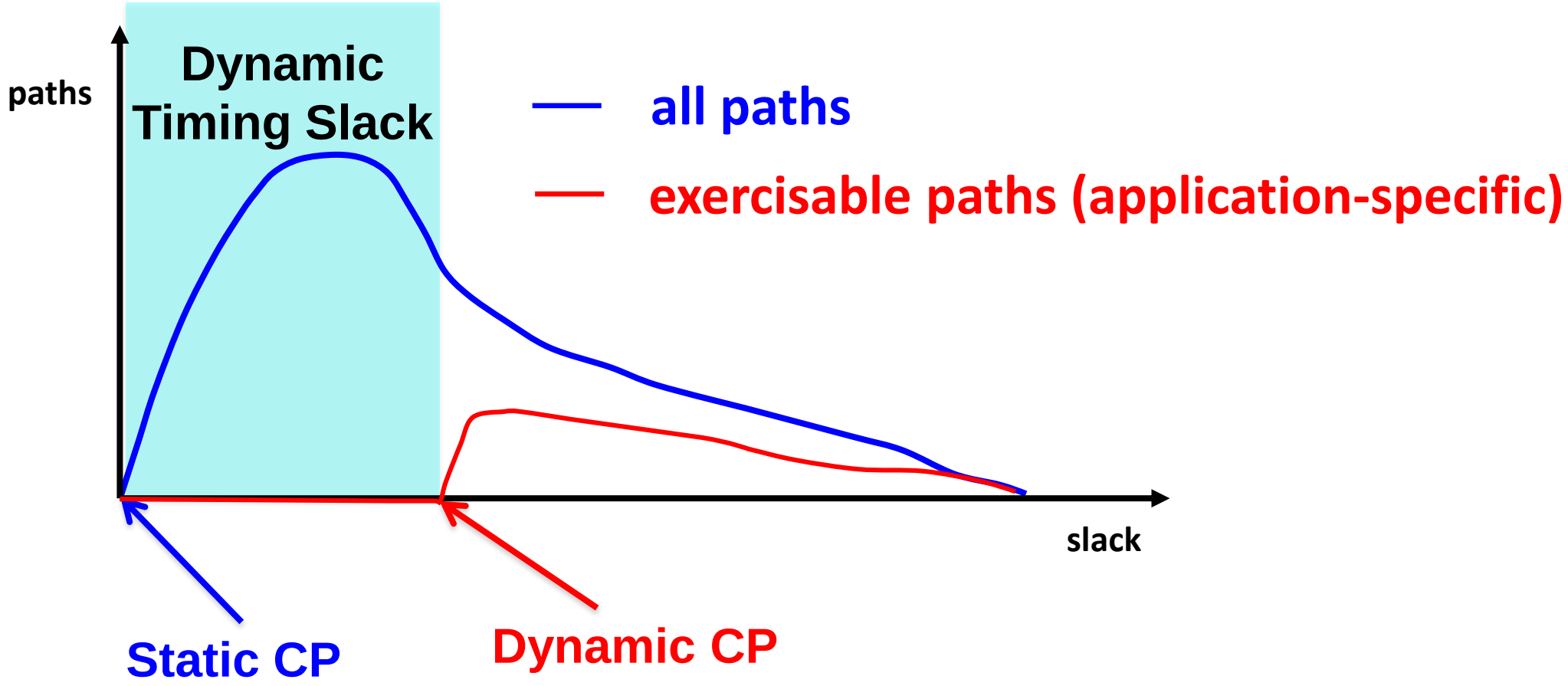
Opportunity



Low-power embedded systems run same application over and over

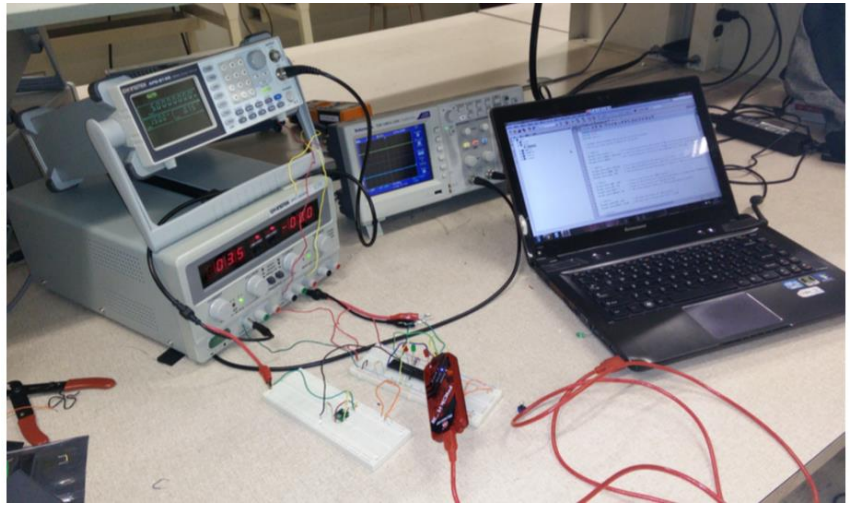
Many applications do not exercise all functionalities of a low power general purpose processor

Dynamic Timing Slack (DTS)



Reduce power (voltage) without reducing performance (frequency)

Dynamic Timing Slack: Fact or Fallacy?



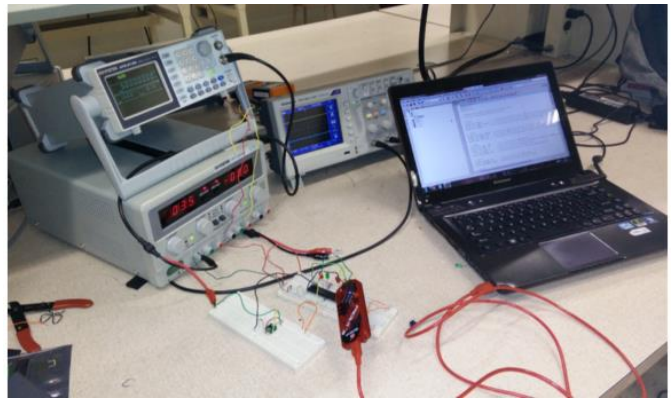
Measurement setup

	PIC24		MSP430	
Benchmark	$V_{min}(V)$	Pwr Saved (%)	$V_{min}(V)$	Pwr Saved (%)
binSearch	1.82	20.2	2.87	30.3
div	1.83	20.3	2.87	33.7
inSort	1.85	17.2	2.90	36.2
intAVG	1.89	13.1	2.77	38.4
intFilt	1.83	20.0	2.92	30.5
mult	1.82	20.4	2.76	41.7
rle	1.77	25.5	2.83	35.9
tHold	1.83	20.1	2.86	34.4
tea8	1.82	20.4	2.82	39.5

Measurements provide evidence of DTS in real embedded systems

How to determine DTS?

Why not use input-based profiling?

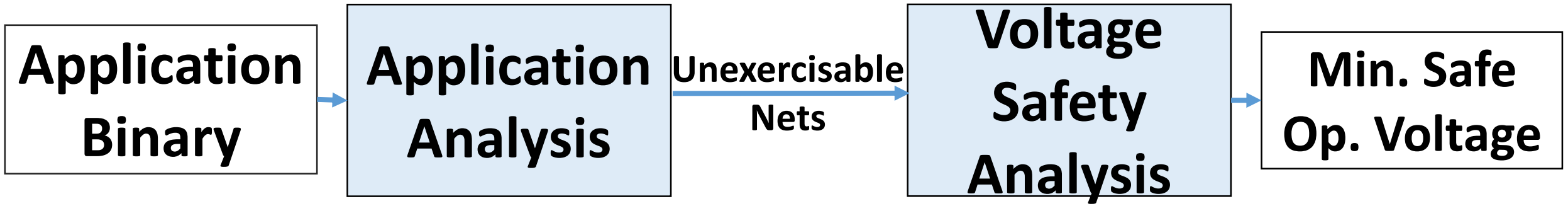


Problem:

- Activity **depends on input**
- Critical path can depend on input
- Variations depend on input and operating conditions

WE NEED A DIFFERENT APPROACH

Identifying Dynamic Timing Slack



Symbolic simulation with X as inputs for input independent analysis

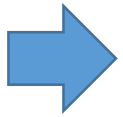
Timing analysis at worst case for variation-independent analysis

Application Analysis

(Binary)

```
1.mov #0, r4;
2.mov #0, r5;
3.mov &0x0020, r15;
4.cmp r15, #10000;
5.jl else
then:
6.mov #1000, r4
7.jmp end
else:
8.mov #1000, r5
end:
9.sub r4, r5, r6;
```

←X

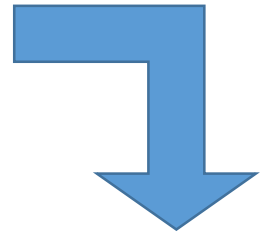


```
1.mov #0, r4;
2.mov #0, r5;
3.mov &0x0020, r15;
4.cmp r15, #10000;
5.jl else
```

```
then:
6.mov #1000, r4
7.jmp end
```

```
else:
8.mov #1000, r5
```

```
end:
9.sub r4, r5, r6;
```



Unexercisable Nets

Voltage Safety Analysis

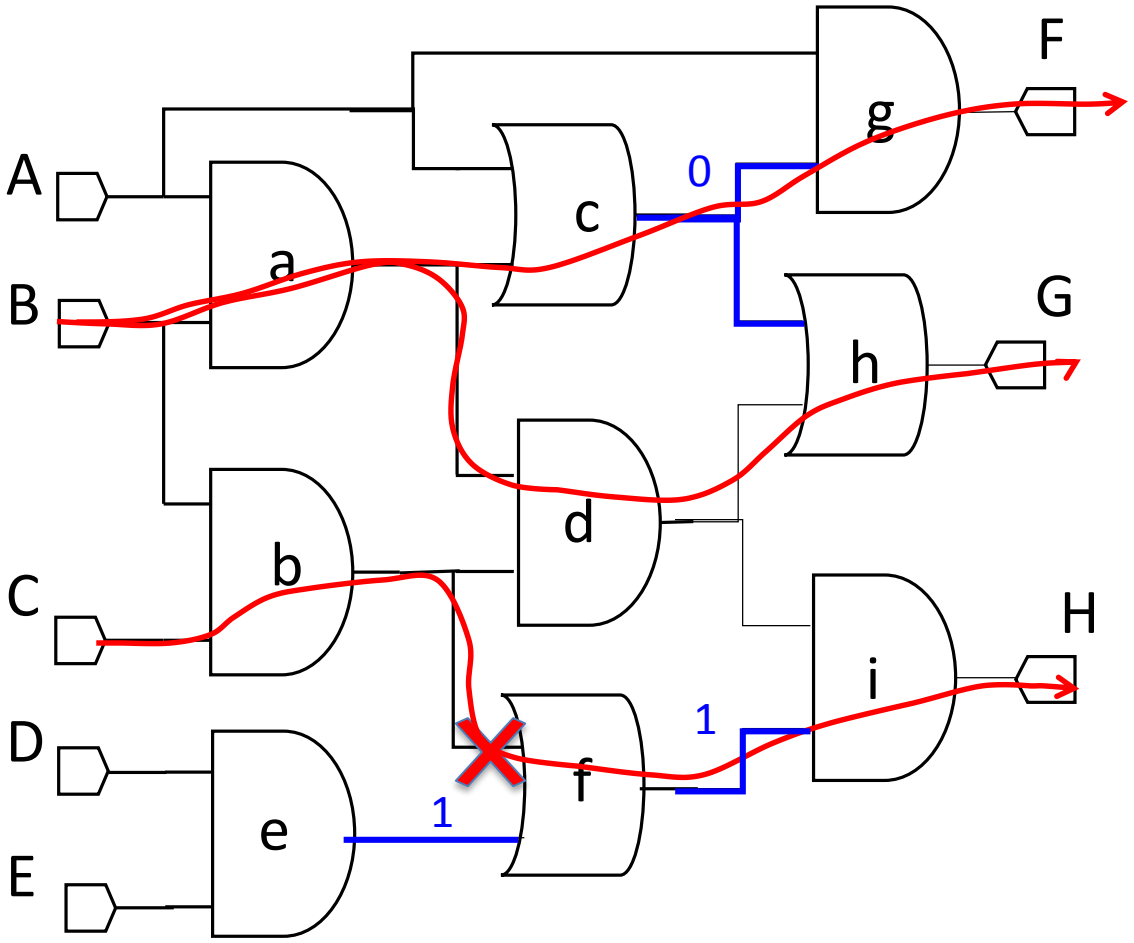
Unexercisable Nets

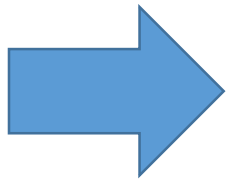
- c -> 0
- e -> 1

Critical Paths

- ~~B-a-c-g-F~~
- ~~C-b-f-i-H~~
- B-a-d-h-G

.
.
.



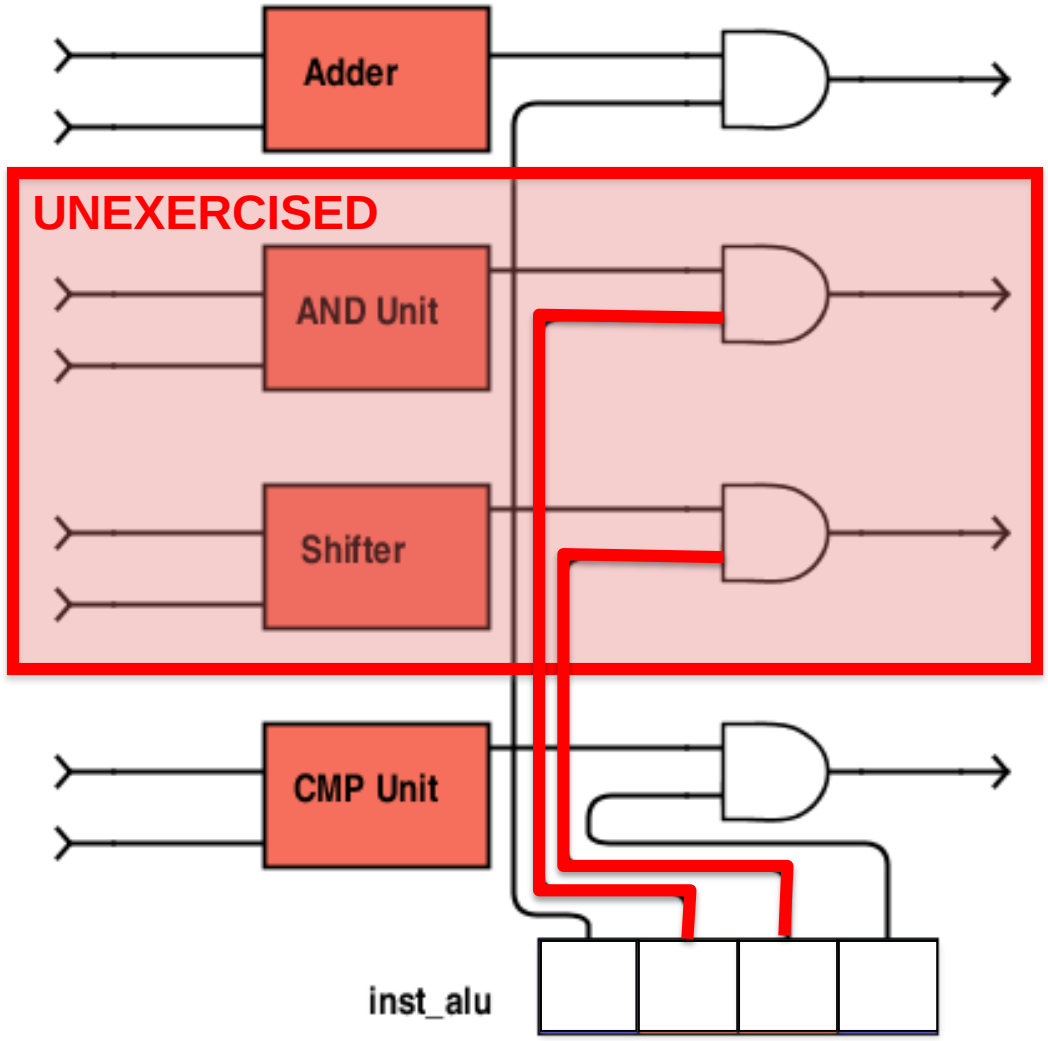


V_{min}

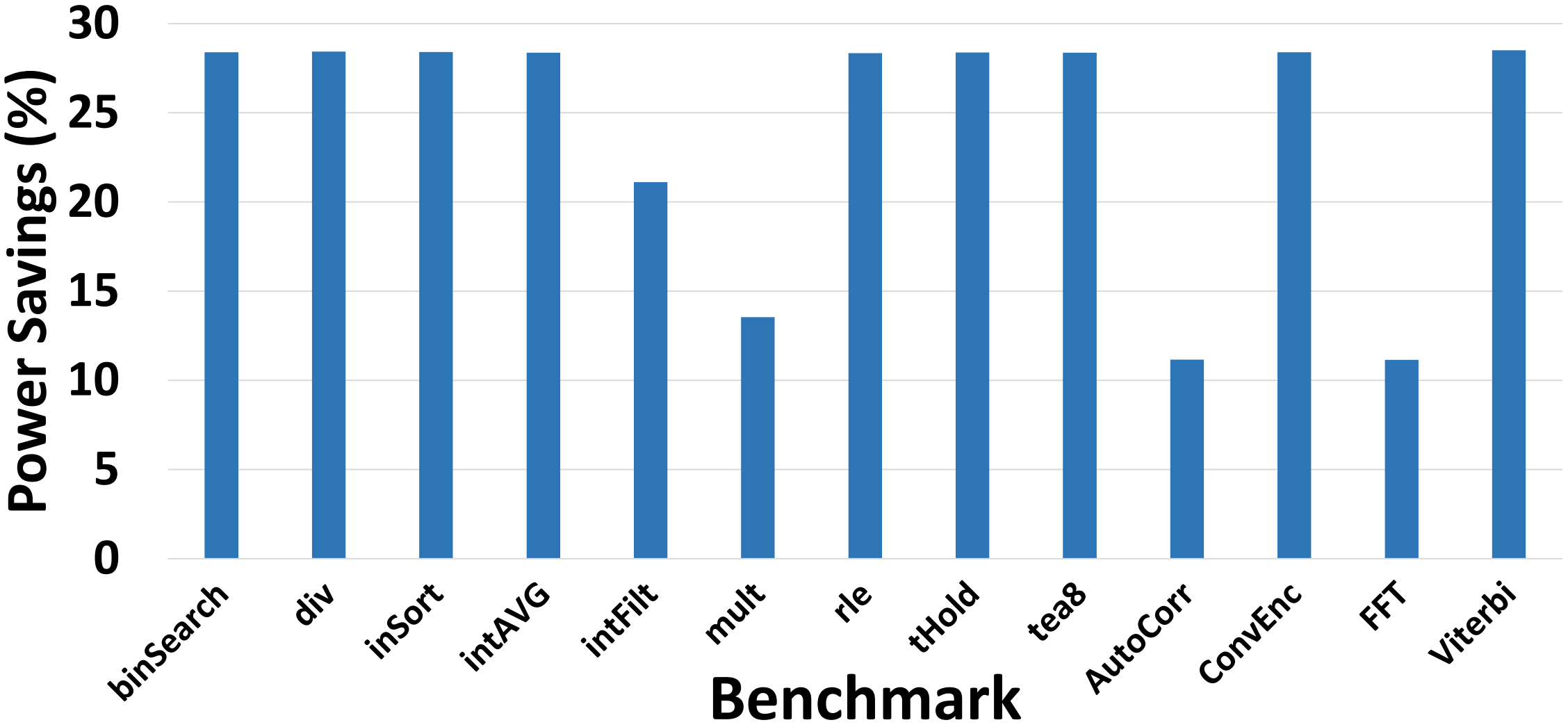
Real Example

example code block for tHold :

```
mov #0, r4 ; Initialize counter
loop:
mov &0x0020, r15 ; Read from mem/port
cmp r15, #10000 ; Threshold Detection
jl else
then:
inc r4
mov r4, 0x0028 ; Write to mem/port
else:
jmp loop
```



Power savings from exploiting DTS



25% free power savings, on average

Conclusion

- Emerging applications are severely power constrained
- Only exercised parts of the processor need to meet timing
- Exploiting Dynamic Timing Slack can **reduce power without reducing performance**
- Average of 25% free power/energy savings