Case Study:

Facial Detection & Recognition for Always-On Applications

Jamie Campbell
Synopsys

SYNOPSYS®

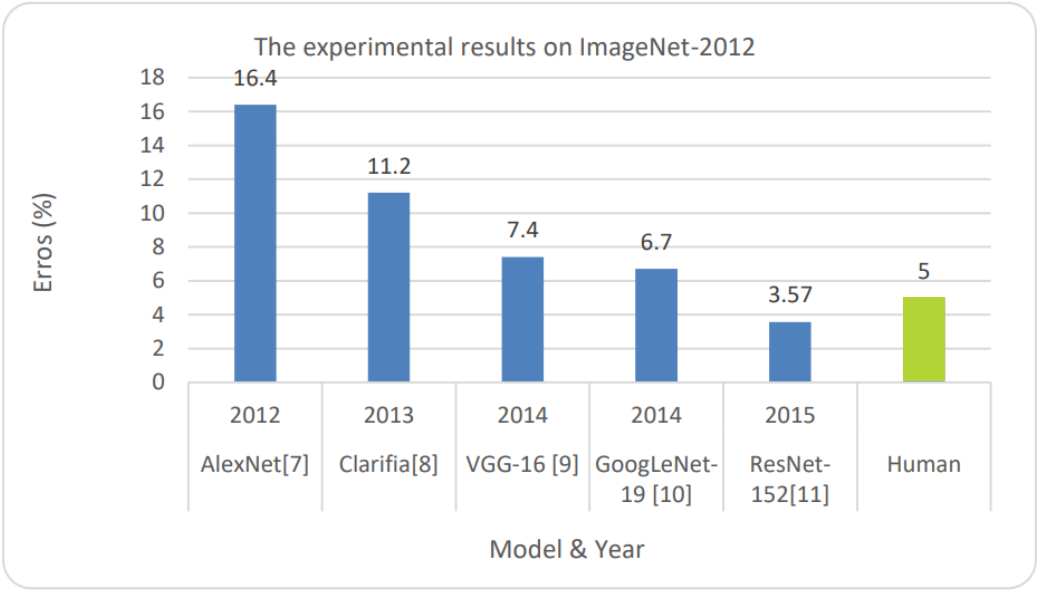
Face Recognition – An Introduction

The challenge of identifying and verifying faces from images:
"Who is this person?" and "Is this the person?"

- Easy task for humans…but much harder for machines
- Useful biometric identification technique
- Advantages
  - Works using inexpensive camera sensors
  - Does not require physical interaction from the user

Face Recognition – Some History

- First attempts to use computers to recognize faces happened in the 1960s
  - Required manual recording of facial features
  - Technology of the time limited developments
- Linear Algebra and the “Eigenface” approach allowed for significant developments in the field
  - Used as a basis for many deep learning algorithms
  - Accuracies improved significantly in the 2000s and 2010s



SYNOPSYS®

Sources: https://anyconnect.com/blog/the-history-of-facial-recognition-technologies & https://en.wikipedia.org/wiki/File:RAND_Tablet.png

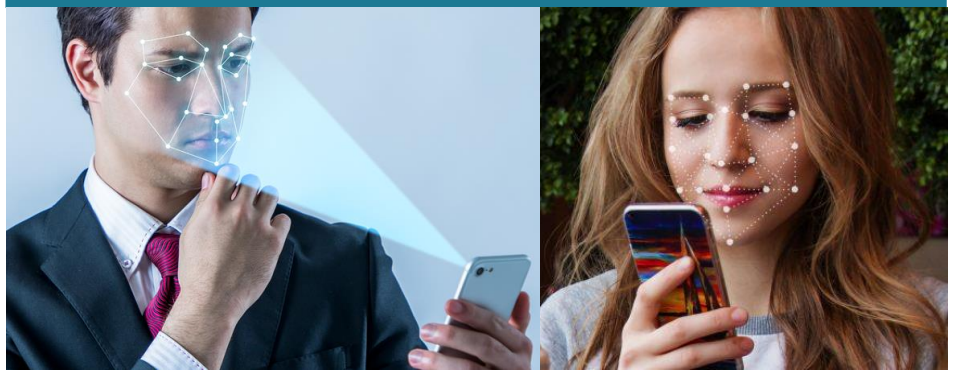
Face Recognition in Embedded Computing

2021 embedded VISION summit

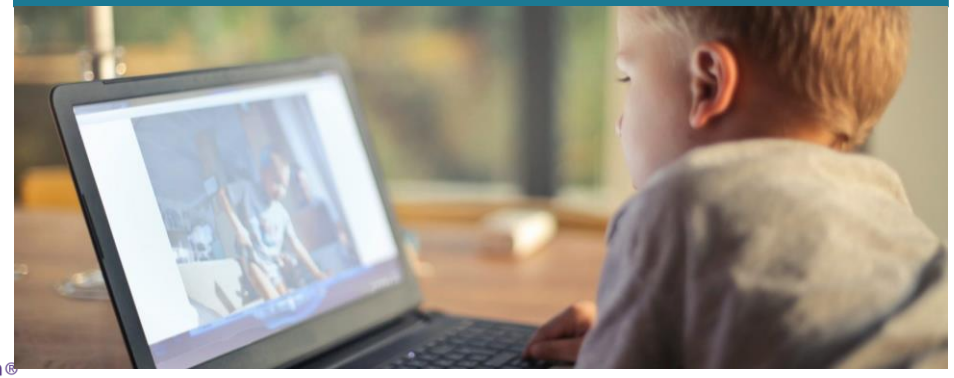
New applications domains demand **on-device** face recognition

Smart Security
Door Locks

Smart Phones



Unlocking Laptops



Payment Devices
Vending Machines, Parking Meters etc

SYNOPSYS®

Embedded Face Recognition System Characteristics

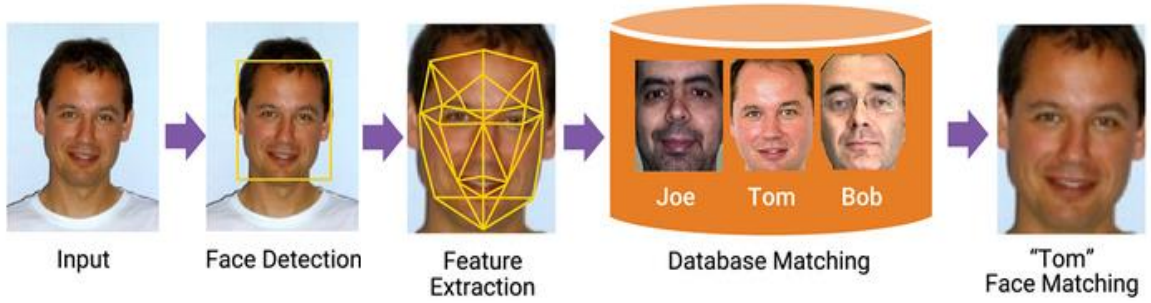
- Embedded Face Recognition systems must
  - ✓ Perform *always-on* monitoring
  - ✓ Be low-power (to support battery-powered scenarios)
  - ✓ Respond to inputs in real-time
  - ✓ Be capable of handling processing requirements of complex face recognition algorithms

DESIGN CHALLENGE

Balance design constraints of an always-on, deeply-embedded device AND performance demands of complex face detection NN networks

Steps to Recognizing a Face

1. Detecting a face: Is there a face in the input image?
2. Locating a face: Where is the face in the input image?
3. Identifying a face: Extract features and match with a database



Each step requires a different amount and type of computation

Can we do this efficiently in an embedded system?

Solution:
Use a phased approach for Detection & Feature Extraction

Phase 1 – Is there a face?  "Yes"  "No"

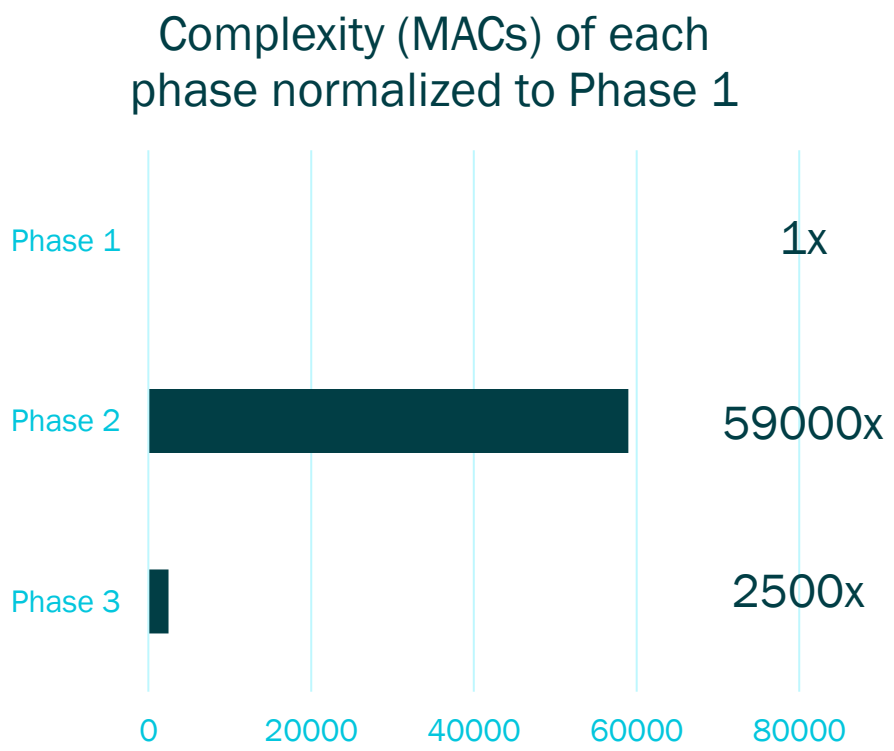
- Low complexity "face detector" NN classifier graph executes continuously
- Uses simple algorithms and efficient hardware to minimize "always-on" energy consumption
- Trigger event: Signal "Yes" detections to next phase

Phase 2 – Confirm there's a face and find it in the frame 

- Execute a high-accuracy face detector NN graph when signaled
- Leverage dedicated NN accelerator to minimize execution time of most complex algorithm

Phase 3 – Compute the "face embedding" vector 

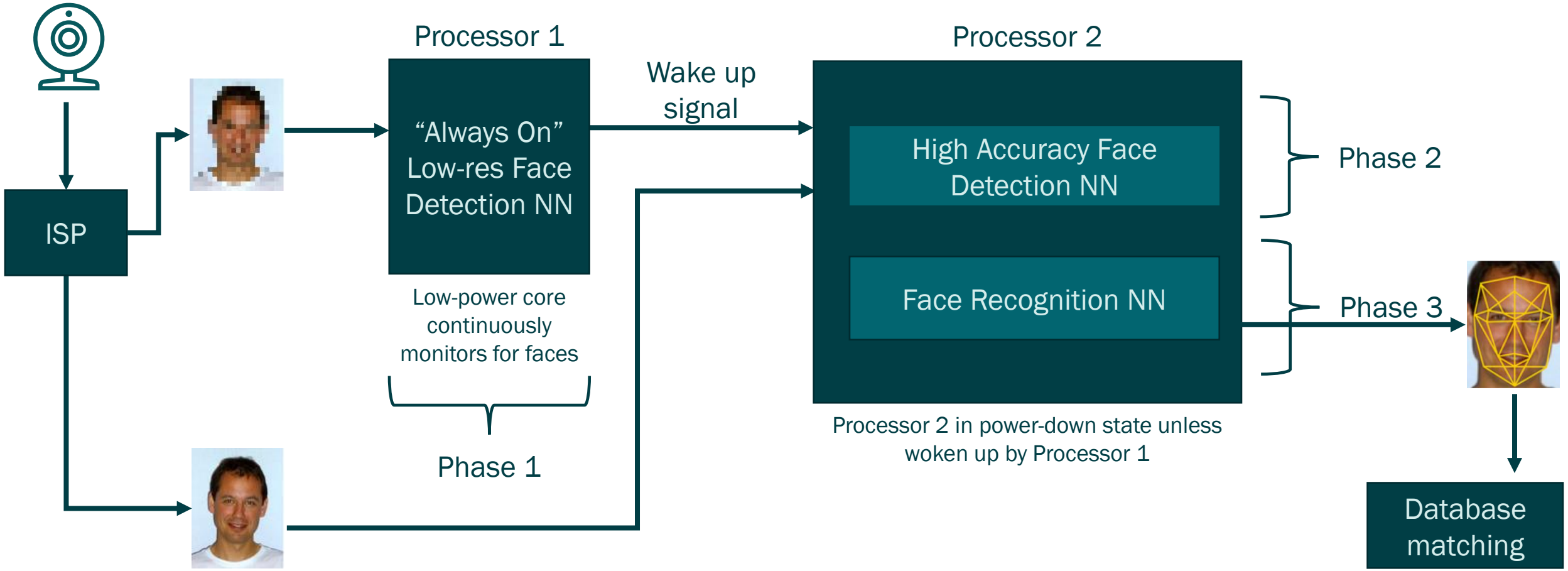
- Vector represents properties of a certain face – can be compared with database



SYNOPSYS®

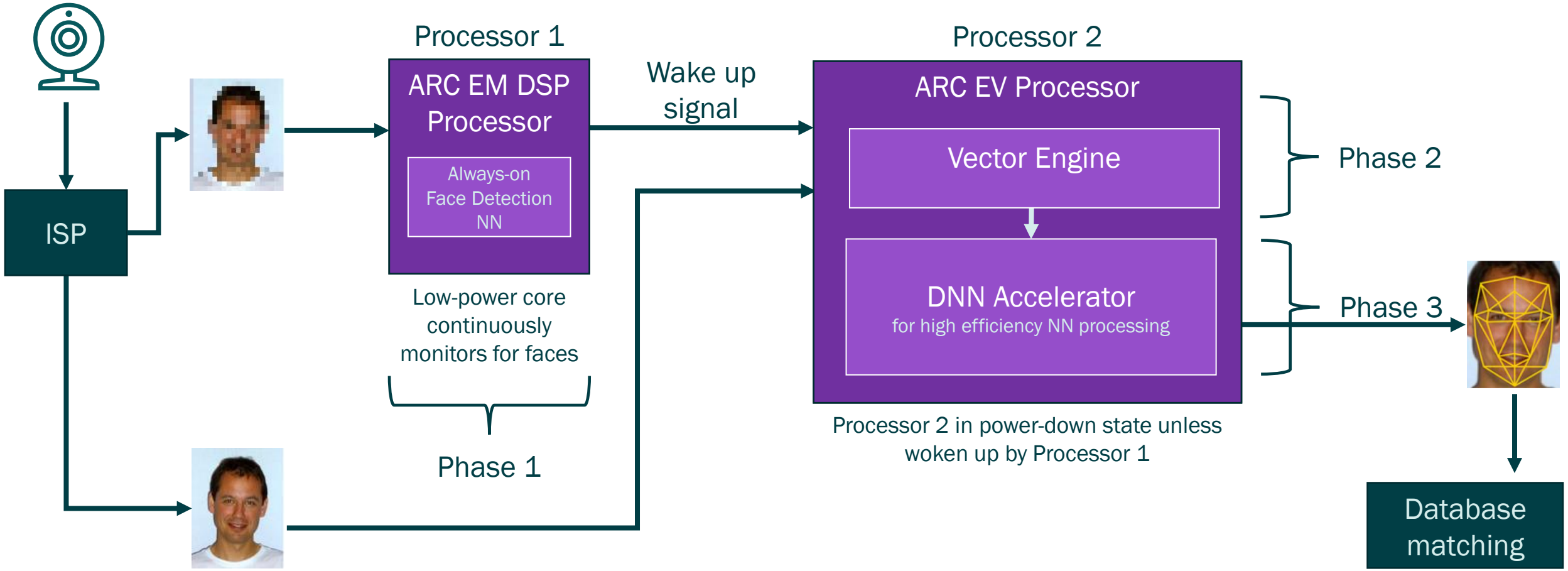
Dividing up the Job

Share the work between a low-power and a high-performance processor to achieve application power targets



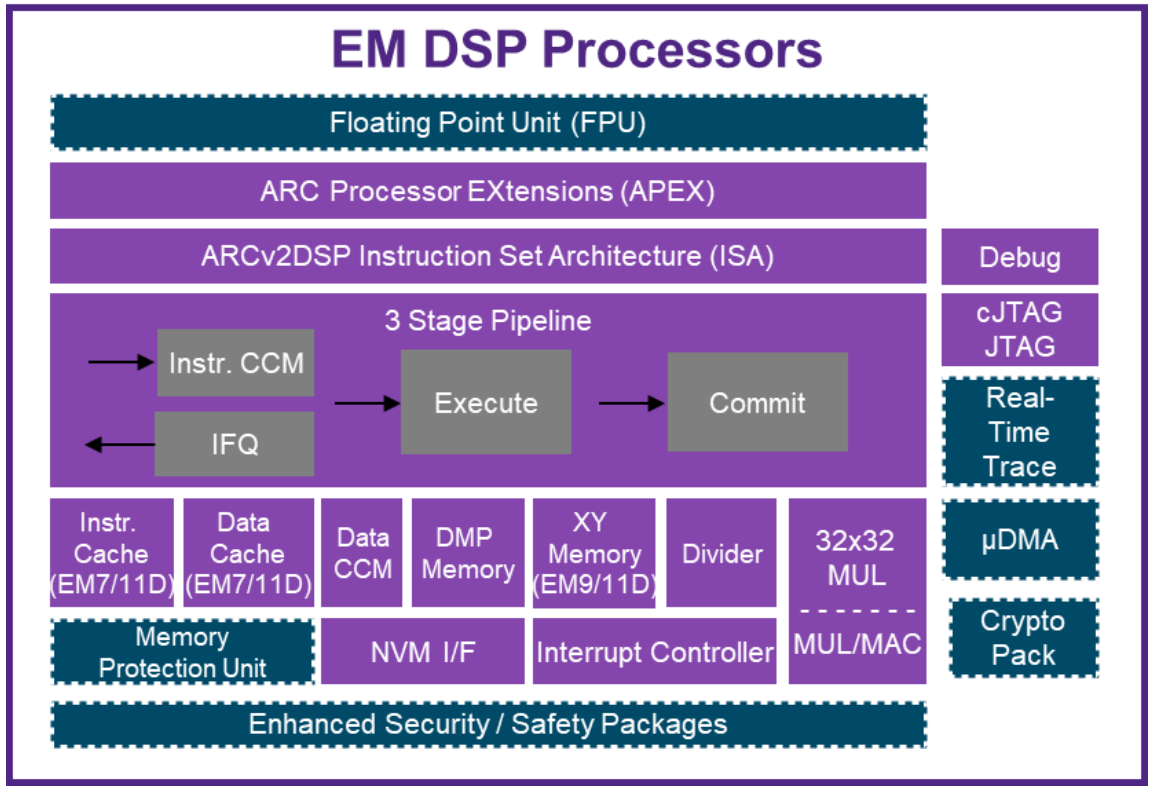
Dividing up the Job

Share the work between a low-power and a high-performance processor to achieve our power targets



Processor 1: Synopsys ARC EM9D

Efficiently Executes Always-On AI Workloads

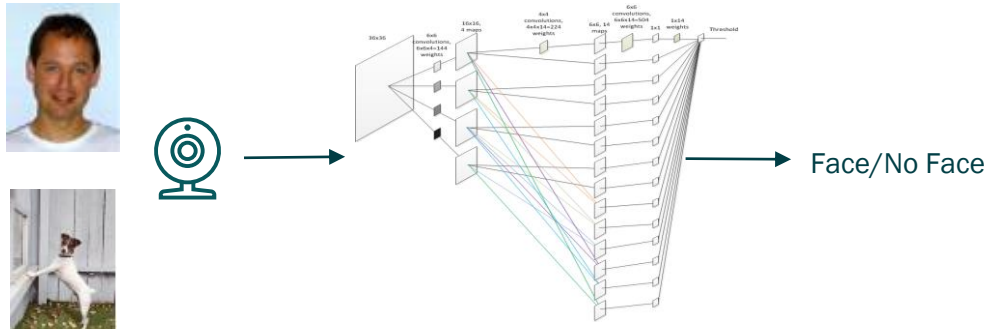


- RISC core with DSP ISA extensions
- Includes key features for efficient DSP/NN processing
  - Vectorized Multiply Accumulate
  - Zero-overhead looping
  - Fast XY memory & address generation units for instruction-level parallelism
- Optimized NN libraries
- RTOS options for more complex control and interface tasks

Phase 1: Simple “Face/No Face” Detection on ARC EM9D

Is there a person looking at the camera?

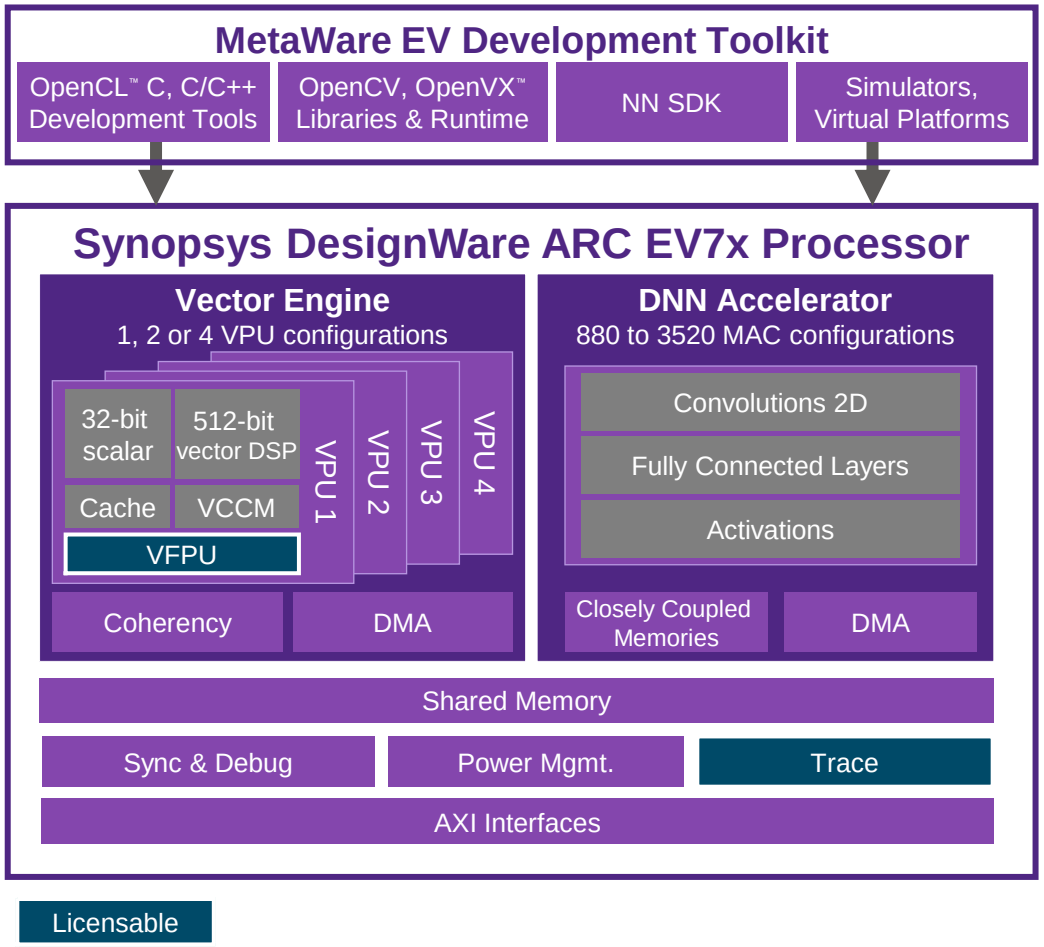
- Simple binary classifier model – “Face” or “No Face”
  - Low res 36x36 input
  - Executes efficiently using optimized NN libraries available for EM9D
- Eg: Run inference 4x per second – ensures real-time response for target application
  - Clock ARC EM processor just fast enough to meet inference rate



4-layer CNN algorithm

Reference: http://parse.ele.tue.nl/system/attachments/11/original/paperspeedsigncnn.pdf

Processor 2: Synopsys ARC EV7x

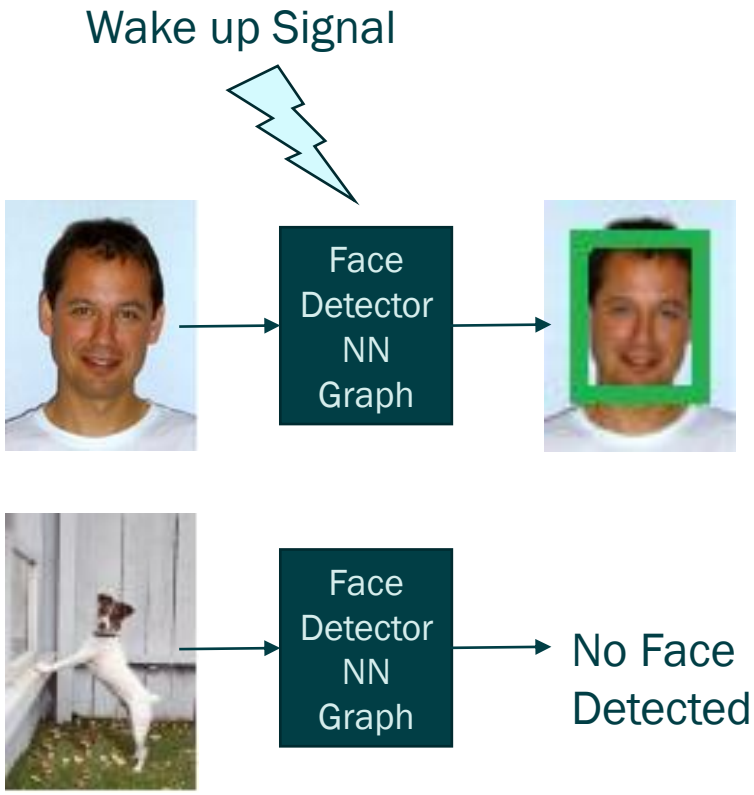


- ARC EV7x Vision Processors include
  - Up to four enhanced vector processing units (VPUs)
  - DNN accelerator with up to 3520 MACs
- Provides scalability for performance vs power tradeoffs
- Designed for maximum power efficiency for neural network processing

Phase 2: Face Detection on ARC EV7x

Confirm there's a face and find it in the frame

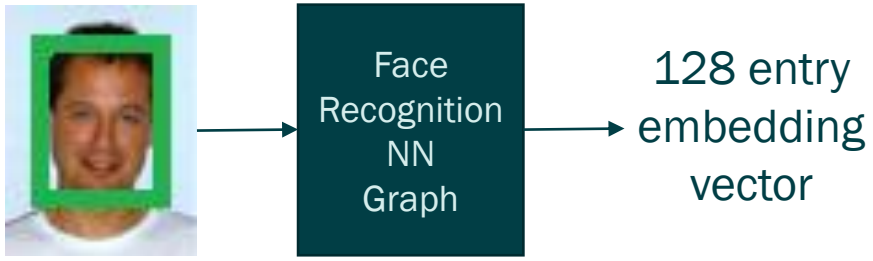
- Runs only after wake-up trigger event from Phase 1
  - Can be real detection or a false positive
  - Eg: Once every 60 secs (very busy door-lock camera + some false detections)
- Localize the face in the camera input frame
  - Eg: MobileNet-SSD graph with 416x416 input
- Most complex phase – needs to execute efficiently
  - Use ARC EV7x DNN accelerator



Phase 3: Face Recognition on ARC EV7x

Compute the “face embedding” vector

- Executed after successful Phase 2 detection
- Extract face embedding vectors suitable for database lookup
  - FaceNet (MobileNetv2-based)
  - Use Phase 2’s detection bounding box as input
- Leverage ARC EV7x’s DNN accelerator
  - Possible to have both graphs loaded at same time



Justifying the Phased Heterogenous Solution for Phase 1

Question: Could we use ARC EV for Phase 1?

Facts

- For both ARC EM and ARC EV, Phase 1 compute energy is not dominant
- Phase 1 workload is exceedingly simple for the ARC EV processor
- ARC EV will be idle most of time (only 4 inferences/second over 60 seconds)

Options for idle ARC EV

1. Sleep mode -> Standby energy
2. Power down -> Boot up energy

Metric	ARC EM always-on	ARC EV sleep when idle	ARC EV power down when idle
Standby Energy Consumption	1x	12x	3x

ARC EM is Highly Energy-Efficient for Always-On Tasks

Justifying the Phased Heterogenous Solution for Phases 2 & 3

Question: Could we use ARC EM for Phase 2 and Phase 3?

Facts

- Phase 2 and 3 compute are complex workloads
- ARC EM-based execution would not be real-time (or would need an unreasonable clock speed to maintain inference rate)
- ARC EV provides significantly more compute parallelization so is more energy-efficient

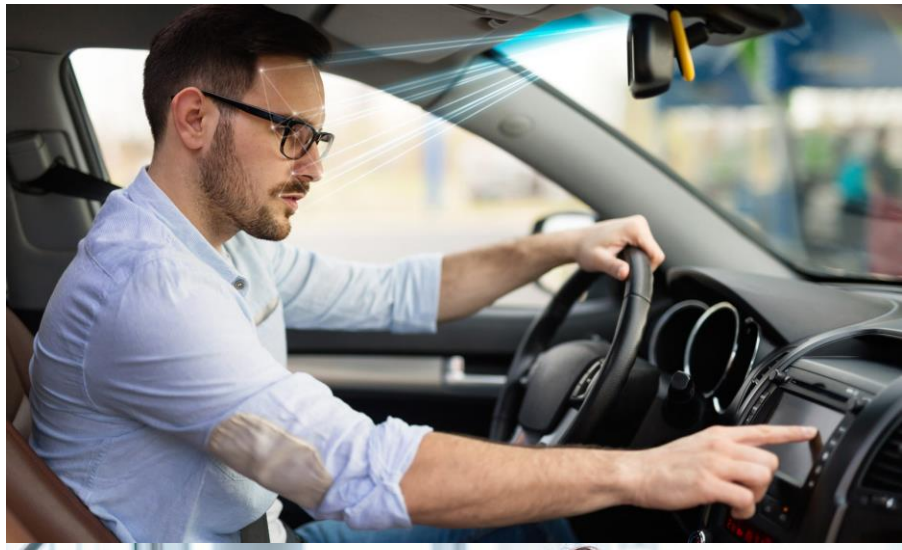
Compute time	EM	EV
Phase 2	410x	1x
Phase 3	185x	1x

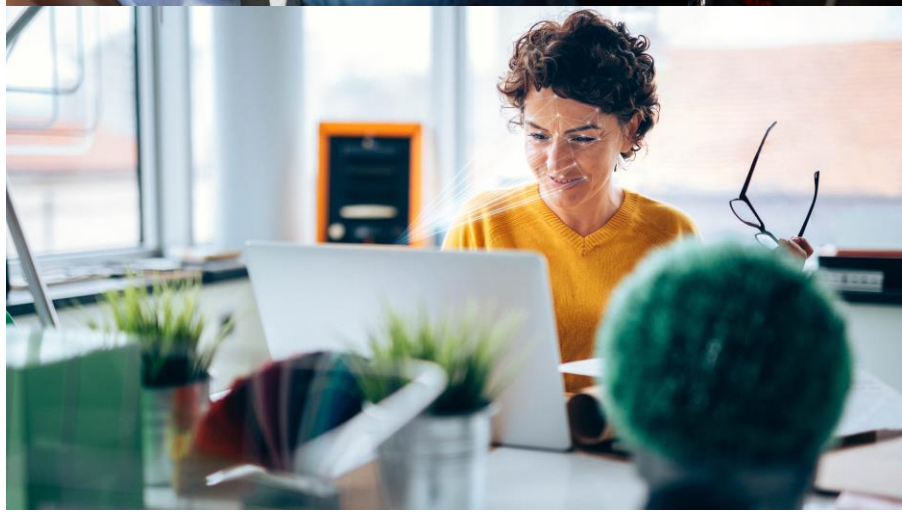
Energy	EM	EV
Phase 2	12x	1x
Phase 3	12x	1x

ARC EV Offers Real-Time Performance and Power Efficiency for Complex Workloads

Conclusion

- Face Recognition is multi-step problem
- For embedded environments, power minimization is key
- Distribute the workload across heterogeneous cores
- Combine strengths of low-power + high-performance cores to achieve energy-saving goals
- Synopsys DesignWare IP offers scalable processors like the ARC EM and ARC EV7x families which are suitable for this work





Resources

Power Efficient Facial Detection & Recognition with ARC Processors
https://www.synopsys.com/designware-ip/technical-bulletin/face-recognition-detection-arc-ev.html

Say Welcome to the Machine. Low-Power Machine Learning for Smart IoT Applications
https://www.synopsys.com/dw/doc.php/wp/arc_low_power_machine_learning_for_iot.pdf

FDDB: A Benchmark for Face Detection in Unconstrained Settings
http://vis-www.cs.umass.edu/fddb/fddb.pdf

The History of Facial Recognition Technologies: How Image Recognition Got So Advanced
https://anyconnect.com/blog/the-history-of-facial-recognition-technologies

2021 Embedded Vision Summit

Demo 1: **SR-GAN Super Resolution on DesignWare ARC EV7x Processors**

Demo 2: **Simultaneous Localization and Mapping Acceleration (SLAM) on DesignWare ARC EV7x Processors**

View and Q&A:

- May 27: 12:00 pm - 1:00 pm PT
- May 28: 10:00 am - 11:00 am PT

Thank You

SYNOPSYS®