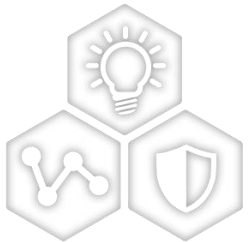


人工智慧嵌入式應用實現 MPU 運行 TensorFlow Lite



A Leading Provider of Smart, Connected and Secure Embedded Control Solutions



SMART | CONNECTED | SECURE

Casper Chang

May-2024

Agenda

- **Artificial Intelligence & Machine Learning**
- **AI Deploying Challenge on MPU**
- **TensorFlow & TensorFlow Lite**
- **AI Applications on MPU Using TensorFlow Lite Framework**
- **Summary**

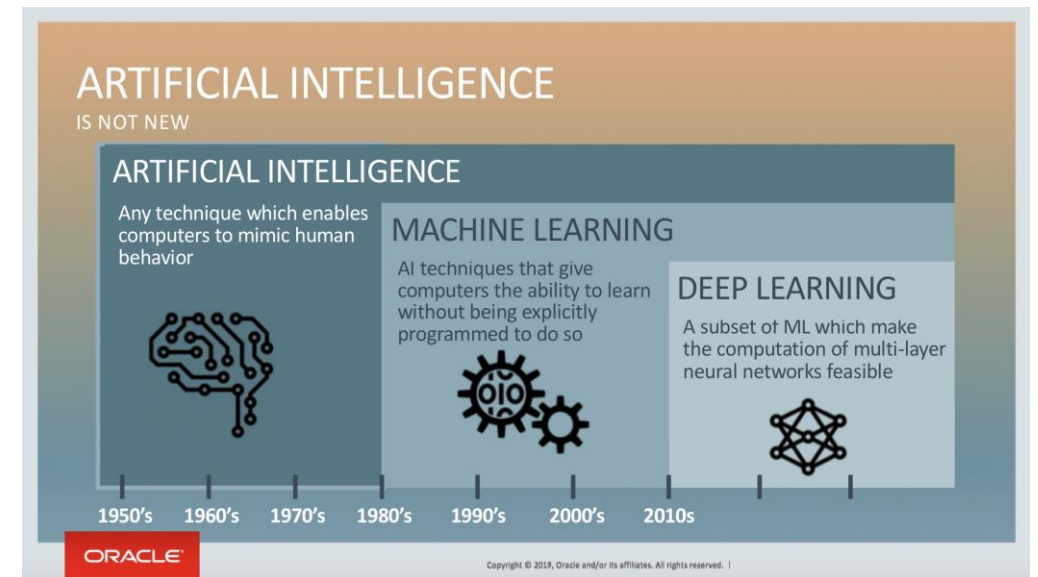
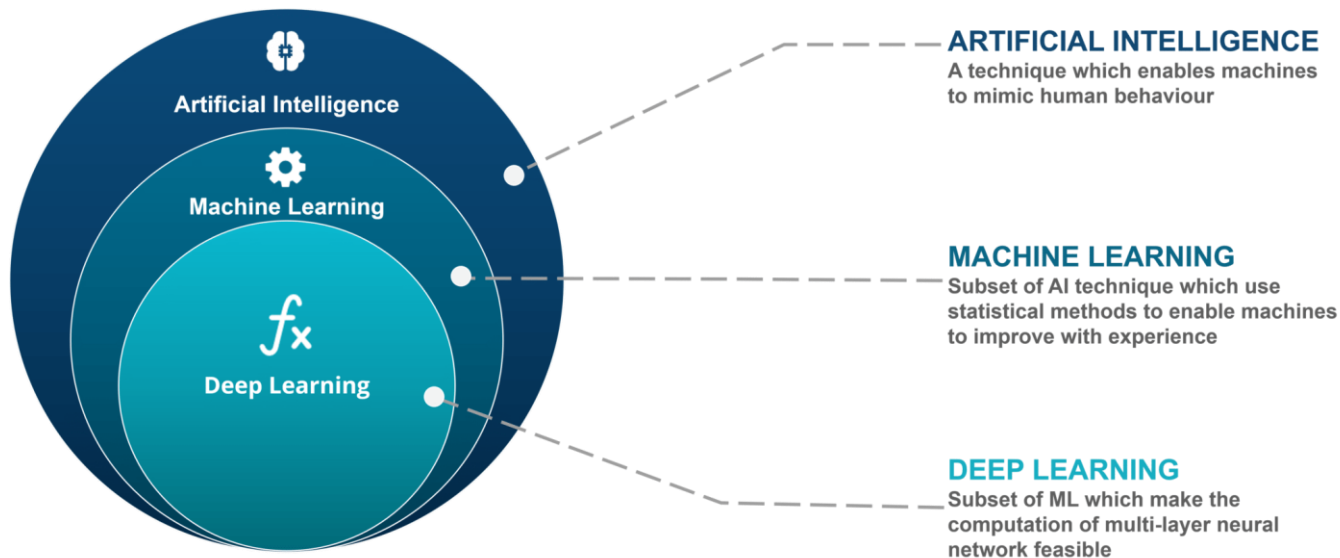
Agenda

- **Artificial Intelligence & Machine Learning**
- AI Deploying Challenge on MPU
- TensorFlow & TensorFlow Lite
- AI Applications on MPU Using TensorFlow Lite Framework
- Summary

What is AI, ML and DL ?

What is artificial intelligence

- A non-human program or model that can solve sophisticated tasks.
- Formally, **machine learning** is a sub-field of artificial intelligence focused on developing algorithms that learn to **solve problems by analyzing data for patterns**. However, in recent years, sometime *artificial intelligence* and *machine learning* are used interchangeably.



<https://blogs.oracle.com/bigdata/difference-ai-machine-learning-deep-learning>

Since the beginning of AI in the 1950s, AI has evolved through its subsets of sciences over the past years to further improve AI capabilities. By today, Deep Learning is one of the most significant techniques for data scientists.

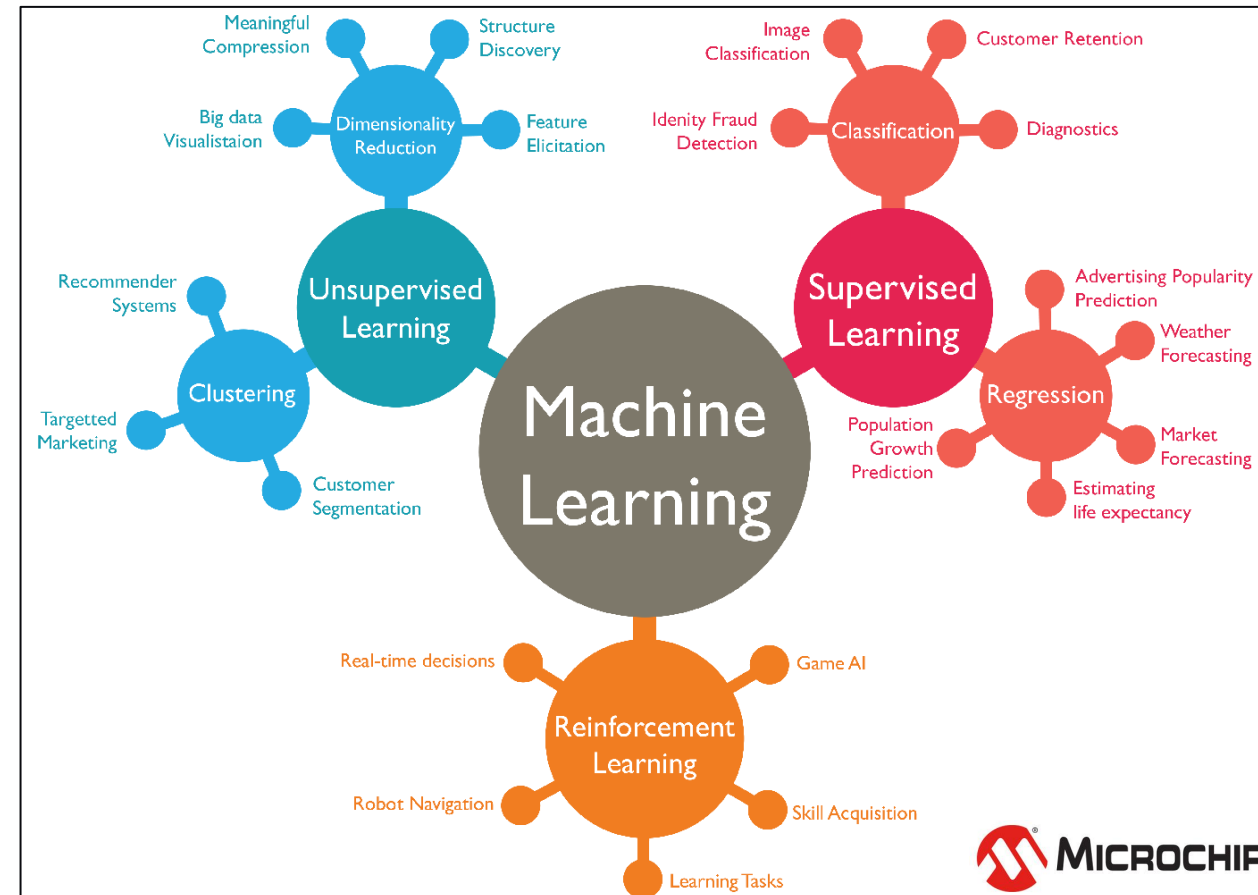
Machine Learning

Different methods for different applications

- Machine Learning is based on three major algorithms: **Supervised Learning**, **Unsupervised Learning** and **Reinforcement Learning**. Each algorithm describes a set of rules and statistical techniques, used to **learn patterns from data**.



- Supervised Learning** (input&output variables were known)
 - Classification of Objects
 - Diagnostics
 - Regression
 - Recommendation
 - Prediction models
- Unsupervised Learning** (input variables were known)
 - Cluster Analysis or Clustering (behavior of groups)
 - Association (people buying product **a**, are maybe buying product **b**)
- Reinforcement Learning** (mostly unlabeled data)
 - Skill acquisition
 - Photo archive (Labeled and unlabeled pictures)
 - Needs a blend learning style with supervised and unsupervised training methodologies.



Agenda

- Artificial Intelligence & Machine Learning
- **AI Deploying Challenge on MPU**
- TensorFlow & TensorFlow Lite
- AI Applications on MPU Using TensorFlow Lite Framework
- Summary

AI Deploying Challenge on MPU

- **Limited computational power**
- **Memory constraints**
- **Power consumption**
- **Real-time performance**
- **Model optimization**
- **Data collection and annotation**
- **Resource management and multitasking**
- **Security and privacy**

AI Deploying Challenge on MPU

Limited computational power

- MPUs typically have limited processing capabilities compared to more powerful hardware like dedicated GPUs or cloud-based systems.
- This limitation can pose challenges when running complex AI models that require significant computational resources.
- Optimizing algorithms and models to fit within the constraints of MPUs becomes crucial.

AI Deploying Challenge on MPU

Memory constraints

- MPUs often have limited memory capacities. AI models, particularly deep learning models, can be memory-intensive, requiring a significant amount of RAM to store parameters, intermediate computations, and data.
- Efficient memory management and model compression techniques are necessary to fit AI models within the available memory.

AI Deploying Challenge on MPU

Power consumption

- MPUs are often used in power-constrained environments such as embedded systems or IoT devices. Running resource-intensive AI algorithms can lead to increased power consumption, which may not be desirable in certain applications.
- Balancing the computational requirements of AI models with power efficiency becomes essential.

AI Deploying Challenge on MPU

Real-time performance

- Some AI applications require real-time or near-real-time inference, where predictions need to be made within strict time constraints. MPUs might face challenges in meeting these performance requirements, especially for complex models.
- Optimizing the model architecture, leveraging hardware acceleration, and employing efficient algorithms are critical to achieving real-time performance.

AI Deploying Challenge on MPU

Model optimization

- MPUs may have specific architectural limitations or support only certain types of operations efficiently. Adapting AI models to exploit the strengths of the MPU architecture and optimizing the model's operations for the available hardware can be a challenge.

AI Deploying Challenge on MPU

Data collection and annotation

- AI models often require large amounts of annotated data for training. Data collection and annotation on MPUs can be limited due to the MPU's limited storage capacity and computational power.
- Addressing this challenge may involve using more efficient data annotation techniques or performing data preprocessing and model training on more powerful systems.

AI Deploying Challenge on MPU

Resource management and multitasking

- MPUs are typically used to execute multiple tasks or applications. In such cases, efficient management and allocation of MPU resources are needed to ensure performance and resource utilization among different applications.
- Additionally, the resource requirements for running AI models need to be considered to avoid conflicts with other tasks.

AI Deploying Challenge on MPU

Security and privacy

- AI applications may handle sensitive data or content with security and privacy risks. Ensuring the security and privacy protection of AI models and data on MPUs is crucial.
- This may involve techniques such as data encryption, access control, and model protection.

Agenda

- Artificial Intelligence & Machine Learning
- AI Deploying Challenge on MPU
- **TensorFlow & TensorFlow Lite**
- AI Applications on MPU Using TensorFlow Lite Framework
- Summary

TensorFlow & TensorFlow Lite, What are they?

- Open-source framework created for training and running machine learning models.
- **TensorFlow Lite** :
 - Part of TensorFlow focused on running machine learning models on mobile and embedded devices.
 - It enables on-device machine learning inference with low latency and small binary size.
 - Addressing 5 **key constraints**:
 - Latency
 - Privacy
 - Connectivity
 - Size (reduced model and binary size)
 - Power consumption
 - Diverse language support, which includes Java, Swift, Objective-C, C++, and Python.





TensorFlow





TensorFlow Lite



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)



TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)

TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)

TensorFlow vs. TensorFlow Lite

Feature	TensorFlow 	TensorFlow Lite 
Functionality	Provides extensive machine learning modeling and training capabilities	Focuses on efficient inference on mobile and embedded devices
Model Optimization	Comprehensive optimization for both training and inference	Emphasizes inference optimization techniques like quantization and hardware acceleration
Deployment Footprint	Relatively larger deployment size	Relatively smaller deployment size
Deployment Platforms	Suitable for various platforms and environments	Designed specifically for the constraints of mobile and embedded devices
Development Workflow	Provides a complete machine learning development workflow	Primarily focuses on model deployment and conversion
Execution Performance	Provides comprehensive training and inference performance	Focuses on achieving efficient inference on resource-constrained devices
Hardware Acceleration	Supports various hardware accelerators and distributed computing	Supports hardware acceleration options like Android NNAPI and Edge TPU
Model Format	Uses TensorFlow model formats (SavedModel, etc.)	Uses TensorFlow Lite model format (.tflite)

Agenda

- Artificial Intelligence & Machine Learning
- AI Deploying Challenge on MPU
- TensorFlow & TensorFlow Lite
- **AI Applications on MPU Using TensorFlow Lite Framework**
- Summary

Introduction to the SAMA7G54

SAMA7G54

Up to 1GHz Performance and 533MHz LPDDR3 Support

- ARMv7 architecture : Cortex-A7
- High Security Features
- Large number of connectivity options
- Complete Imaging and Audio Sub-system
- Optimized BGA ball-out facilitates 4-layer PCB designs
- Simplified external power management and optimized PMIC



Industrial



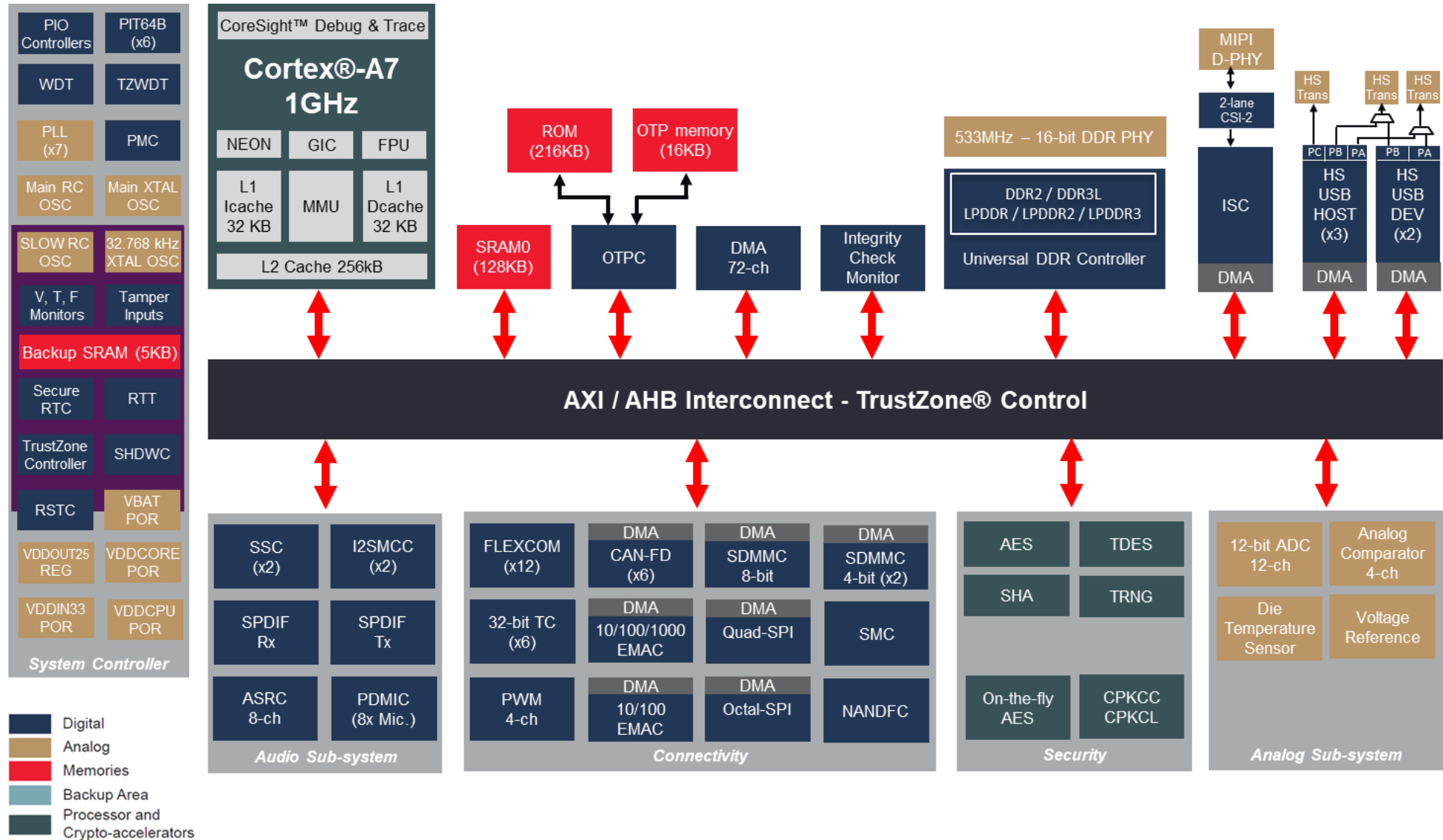
Gateway

SAMA7G54

Up to 1GHz
533MHz DDR3
MIPI Camera
10/100 Ethernet
Giga Ethernet
Security
Audio
6x CAN-FD
QSPI, Octal SPI
Up to 136 I/O
AEC-Q100



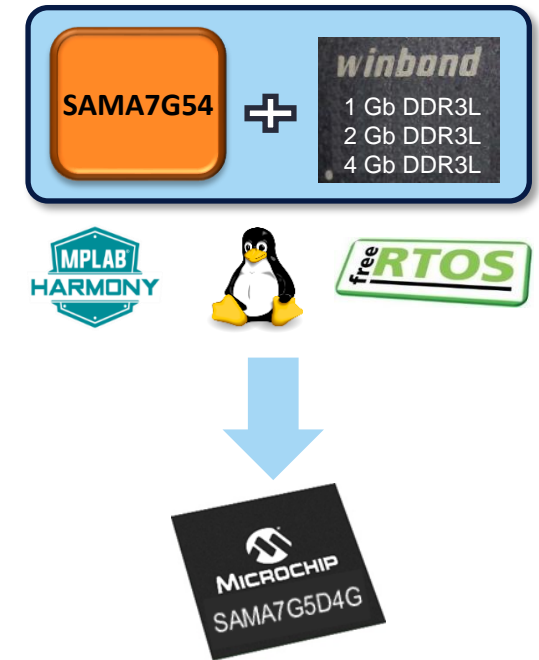
SAMA7G54 Block Diagram



SAMA7G54DxG

System In Package (SiP)

- **SAMA7G54DxG SiP**
 - SAMA7G54 + 1Gb, 2Gb or 4Gb DDR3L
 - *Pin-compatible*: BGA427, 21x18mm, 0.8mm pitch
- **Combines MPU + DDR in a single package**
 - Microchip: Single supplier for MPU & Memory
 - Simplified PCB layout and *reduced EMI risks*
 - Smaller footprint
 - Optimized for 4-layer PCB design *lowers system cost*
 - Removes DRAM obsolescence and price fluctuation
 - Industrial grade (-40/+85°C)



System In Package	Target Application	Memory			EAS
		Type	Density	Bus Width	
SAMA7G54D1G	Linux	DDR3L	128MB / 1Gb	16-bit	Jul'23
SAMA7G54D2G	Linux		256MB / 2Gb		Available
SAMA7G54D4G	Linux		512MB / 4Gb		Jul'23

TensorFlow Lite and the SAM7G54

- **The SAM7G54 has a Cortex-A7 CPU.**
- **This CPU embeds a Floating-Point Unit (FPU)**
- **The Cortex-A7 FPU features are:**
 - Support for single-precision and double-precision floating-point formats
 - Support for conversion between half-precision and single-precision
 - Support for *Fused Multiply Accumulate* (FMAC) operations
 - Normalized and denormalized data are all handled in hardware
 - Trapless operation enabling fast execution.
- **TensorFlow Lite Python package is compatible with the MPU ARMv7 architecture and uses the FPU for full optimization.**

AI/ML Inference Models Run on SAMA7G54

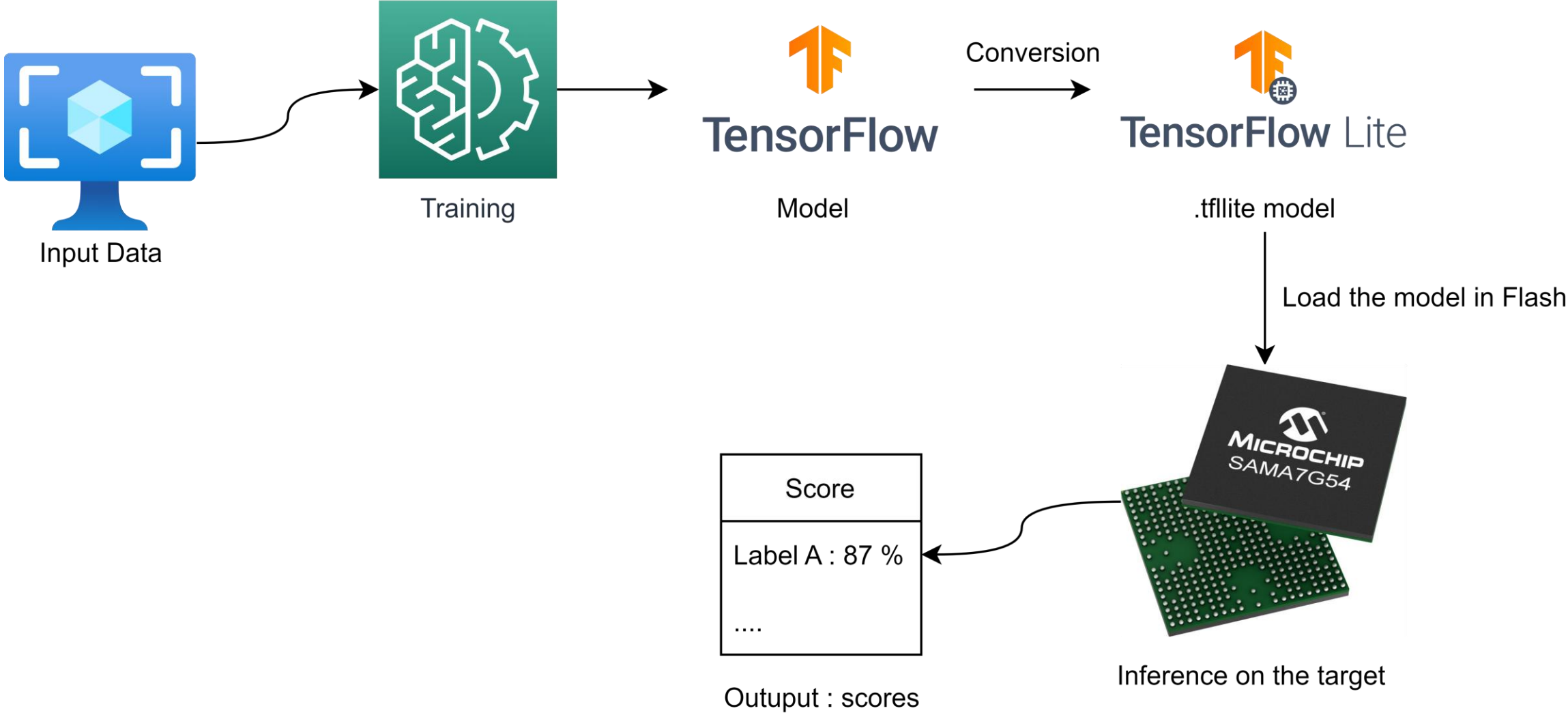
Pre-Trained Inference Models Require Much Less Performance

- Machine Learning follows similar process for most applications



- **TensorFlow Lite is an open-source framework for running pre-trained inference models at the edge**
 - Runs with low latency on MCUs/MPUs
 - Runs in Python in Linux on MPUs

Tflite_runtime Execution Workflow



Building right Linux® Image

- Dedicated Linux4SAM distribution provided :
- Linux4SAM 2023.04 (New: 2023.10)
- Customized to address tflite_runtime dependencies constraints :
 - Python Version changed to 3.9.9
 - to address tflite_runtime dependencies
 - Added support for some other libraries like SciPy (Signal processing library)
- <https://www.linux4sam.org/bin/view/Linux4SAM>



AI Applications Demos on MPU

Under a Linux Environnement

Object Recognition Demo

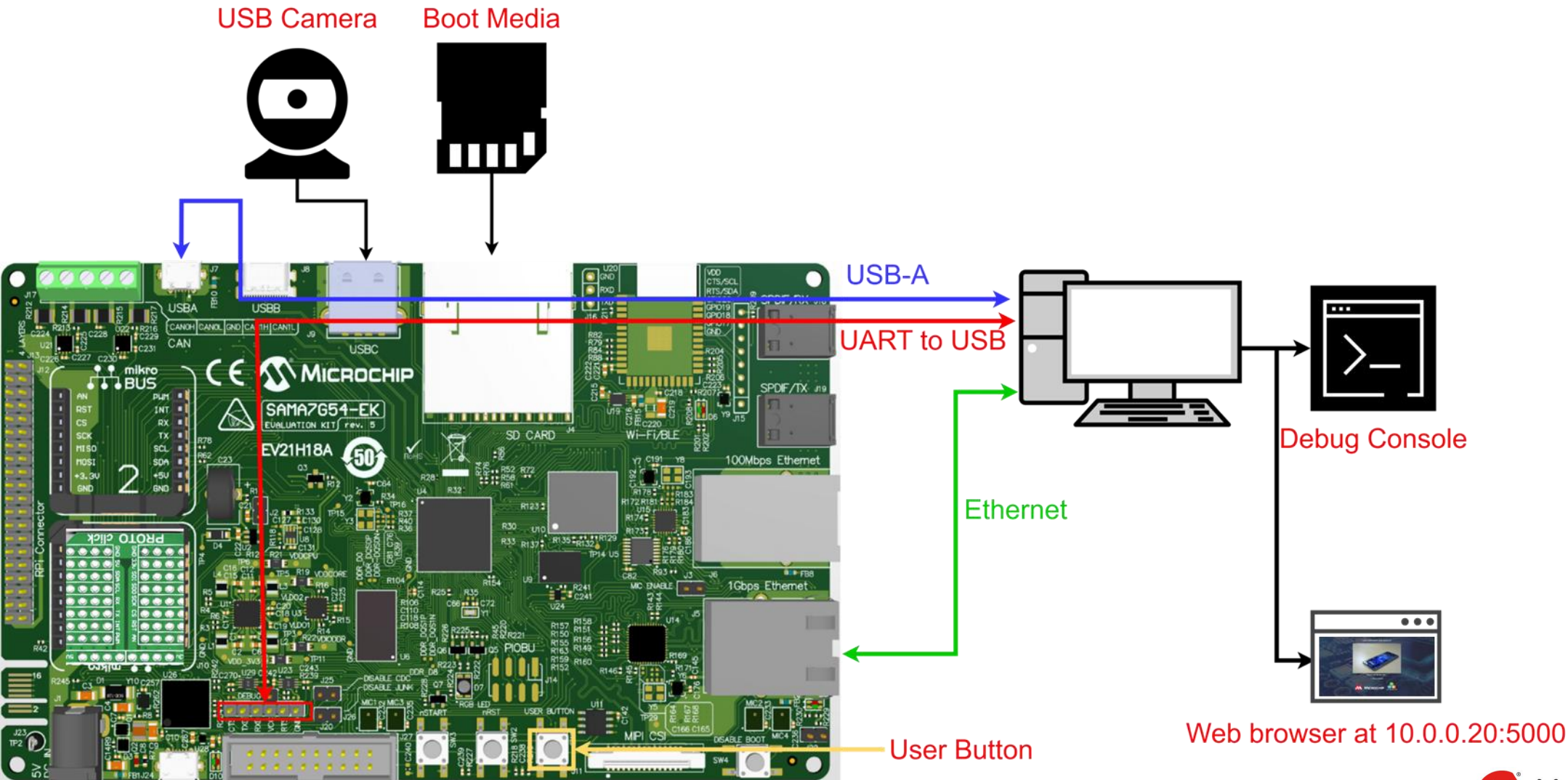
Object Recognition Demo

- Working on SAMA7G54-EK board
- Use of a USB Camera or a MIPI CSI-2[®] Camera
- Three versions :
 - Perpetual recognition performed in a loop
 - Recognition triggered by pressing the user button (recommended)
 - Recognition performed on a static image
- Embedded webserver for video streaming and result displaying



SAMA7 Demo Set-up and Output

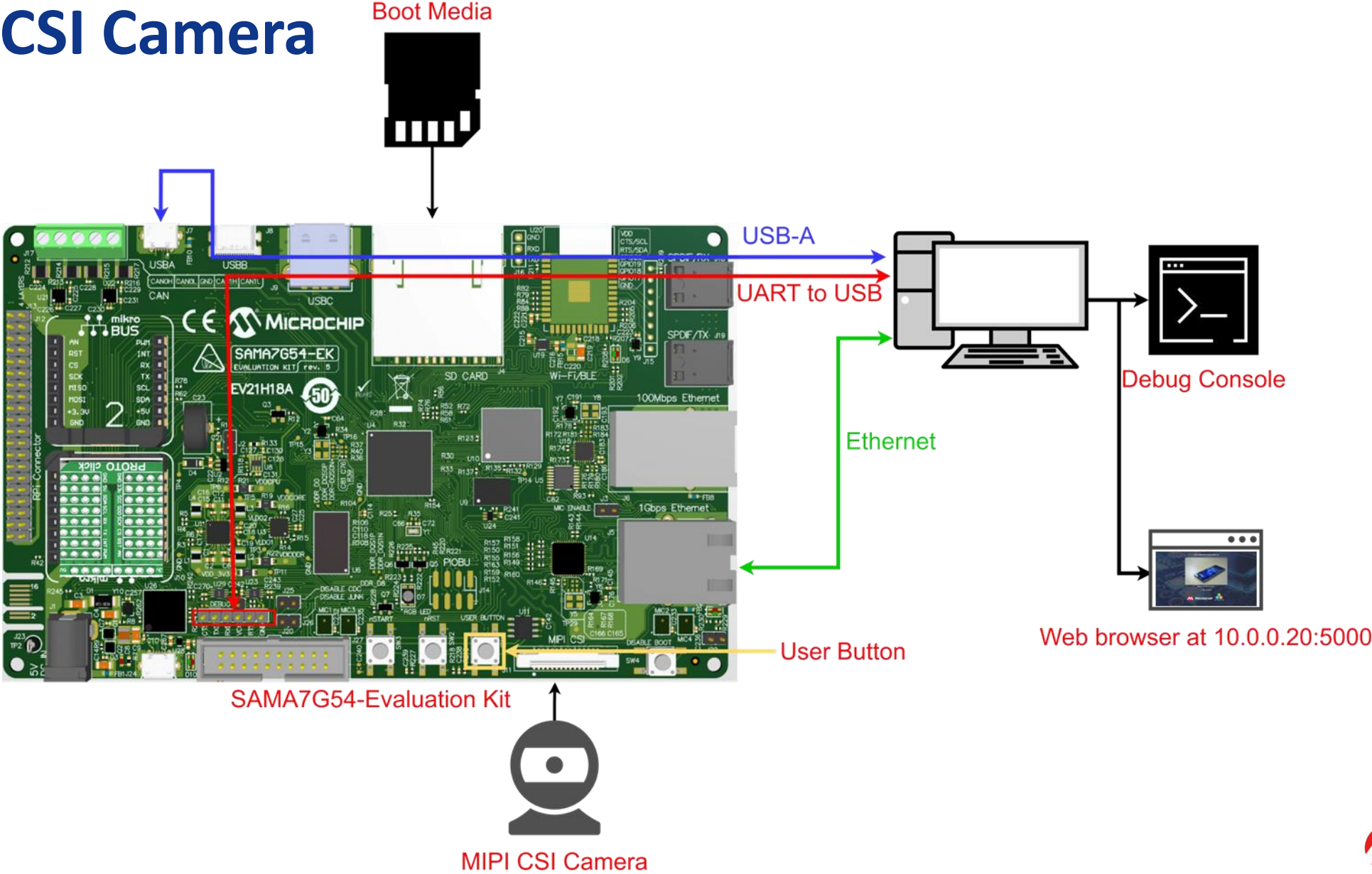
USB Camera



SAMA7G54-Evaluation Kit

SAMA7 Demo Set-up and Output

MIPI CSI Camera




Object Detection Model

- **Model : COCO SSD MobileNet v1**
 - SSD : Single Shot Detector => The image is processed only once through the model
 - Model trained on the COCO Dataset (Common Objects in COntext)
 - MobileNet v1 architecture.
 - Efficient, low-power and low-latency architecture
 - Competitive accuracy.
 - Well-Suited for the Embedded world
 - 79 Objects
 - For example : person, car, dog, bench, etc




Object Recognition Demo

LIVE STREAMING AND RESULTS




OBJECTS DETECTED :

PRESS [USER BUTTON] TO START PROCESSING



MICROCHIP



SMART | CONNECTED | SECURE

Automatic License Plate Recognition (ALPR) Demo

ALPR Demo

- **Working on SAMA7G54-EK board and SAMA5D27-WLSOM-EK board.**
- **Use of a USB Camera**
- **Two steps => Two models :**
 - License plate Detection :
 - Recognition of the license plate on an image, and crop-it.
 - Optical Character Recognition (OCR) :
 - Image cropped around the license plate =>Read the characters
- **Embedded webserver for video streaming and result displaying**

ALPR: Used Models

- **Two steps => Two models :**

- License plate Detection :

- Recognition of the license plate on an image, and crop-it.

- MobileNet-SSD :

- **Model** used predominantly for **object classification** for mobile and embedded vision applications
- MobileNets: pre-trained, low-power models, small, low-latency
- MobileNets can be run efficiently on SAMAX devices with TensorFlow Lite

- Optical Character Recognition (OCR) :

- Image cropped around the license plate =>Read the characters

- LRPNet :

- LRPNet for License Plate Recognition Net
- Model used for sequence prediction for license plates characters.
- Takes a license plate image as input.

Running the application - ALPR (Webserver)

- The Application captures images from the USB Camera.
- These images are used as inputs for the first model which returns a cropped image with only the license plate.
- This second image is used as input for the second model (OCR).
- The output is the predicted sequence.

The screenshot displays the Microchip ALPR demo webserver interface. At the top, the Microchip logo and the text "WELCOME TO THE ALPR DEMO" are visible, along with a note: "MADE WITH LOVE BY THE MPU32 MARKETING TEAM. SCROLL DOWN TO WATCH THE STREAM". The main content area shows the "DETECTED LICENCE PLATE : REH5678". Below this, there are two sections: "RESULT" and "LIVE STREAMING". The "RESULT" section shows a small image of the license plate with a yellow bounding box around it, labeled "CAPTURE & PLATE DETECTION". The "LIVE STREAMING" section shows a larger image of the license plate with the text "REH-5678" overlaid. At the bottom of the interface, the Microchip logo is repeated, along with the slogan "SMART | CONNECTED | SECURE" and three icons representing smart, connected, and secure features.

Running the application - ALPR (Debug Console)

- The Application captures images from the USB Camera.
- These images are used as inputs for the first model which returns a cropped image with only the license plate.
- This second image is used as input for the second model (OCR).
- The output is the predicted sequence.

```
# ./menu_demos_lan
*****
*** Welcome to the SAMA7G54-Ek Demos selection menu ***
*** This menu is made to run demos in a LAN ***
*** Made with love by the MPU Marketing Team ***
*****
1) Object detection demos
2) Keyword recognition demos
3) Automatic License Plate Recognition demo
4) Food classification demo
5) Quit

Press [ENTER] to display the options if they are not shown
Please choose an option :
3

*****
*** Welcome to the SAMA7G54-Ek ALPR demo ***
*** Using a USB Camera ***
*** Made with love by the MPU32 Marketing Team ***
*** Feel free to contact us if needed ***
*****

CONNECT TO : 10.0.0.20:5000
* Serving Flask app "video_stream_flask" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
input_height : 240
input_width : 320
LPR shape : [ 1 24 94 3]
['<Beijing>', 'R', 'E', 'H', '5', '6', '7', '8']
input_height : 240
input_width : 320
LPR shape : [ 1 24 94 3]
['<Beijing>', 'R', 'E', 'H', '5', '6', '7', '8']
input_height : 240
input_width : 320
LPR shape : [ 1 24 94 3]
['<Beijing>', 'R', 'E', 'H', '5', '6', '7', '8']
```

Food Classification Demo

Food Classification Demo

- Working on SAMA7G54-EK board
- Use of a USB Camera
- Recognition triggered by pressing the user button
- Embedded webserver for video streaming and result displaying
- **Model : aiy_vision_classifier_food_V1_1**
 - The architecture of the model is a MobileNet v1 architecture
 - Model trained to recognize **2024 dishes** around the globe.
 - Model comes from TensorFlow Hub.



Food Classification Triggered by Pressing a Button

- The Application reads frames from USB camera and turns these frames in images processed through the model.
- Outputs values are indexed to the appropriate label and the probability that the image match the label
- Press the button to capture the image and to process it through the model.
- The application supports a webserver to display the livestream and the result.
- Connect to the indicated IP address to view the webpage.

```
*****
*** Welcome to the SAMA7G54-Ek Food Recognition demo ***
*** Using an USB Cam and the user button ***
*** Made with love by the MPU32 Marketing Team ***
*** Feel free to contact us if needed ***
*****
usb 1-3: reset high-speed USB device number 2 using atmel-ehci
CONNECT TO : 10.159.226.145:5000
* Serving Flask app 'video_stream_flask'
* Debug mode: off
Module loaded... running interpreter
***Waiting for a button press to capture image***
***Make sure you have browsed to the webpage***

***Button press detected***
img ok
Food detect Inference time: 3855.319ms
Spaghetti with 33.17%
Squid with 17.96%
Mortadella with 16.29%
Carbonara with 16.29%
Ragout with 16.29%
***Waiting for a button press to capture image***
```



Reference: wiki

Simple Audio Recognition Demo

Simple audio recognition: Recognizing keywords

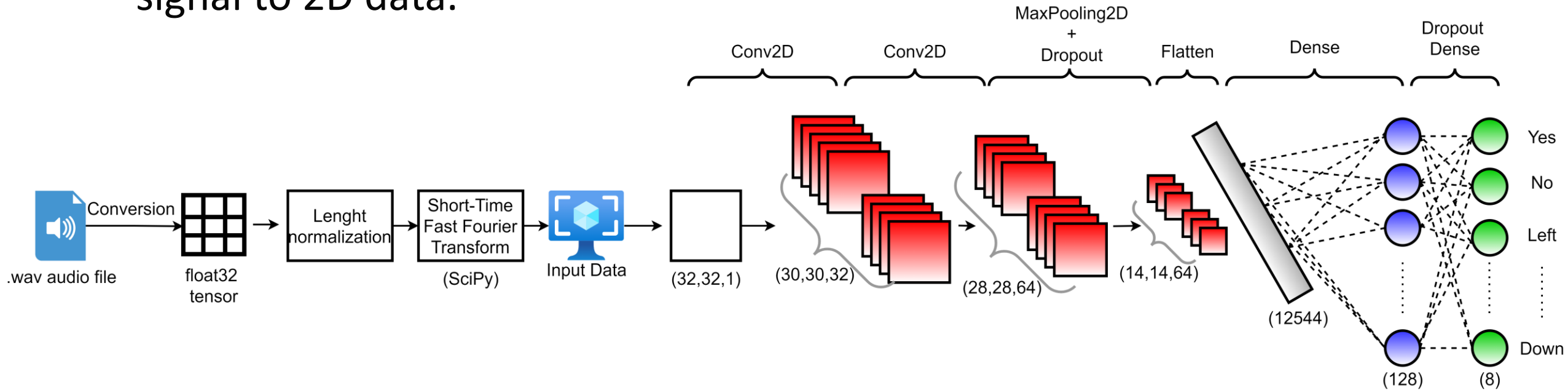
- Keyword recognition demo
- On SAMA7G54-EK Board
- Keywords : {**Left, Right, Up, Down, Yes, No, Go, Stop**}
- Using PDMC0 as a “MIC”.
- Recording via Alsamixer
- Two versions :
 - Dynamic version using the MIC
 - User-button launched version



Audio Keyword recognition model

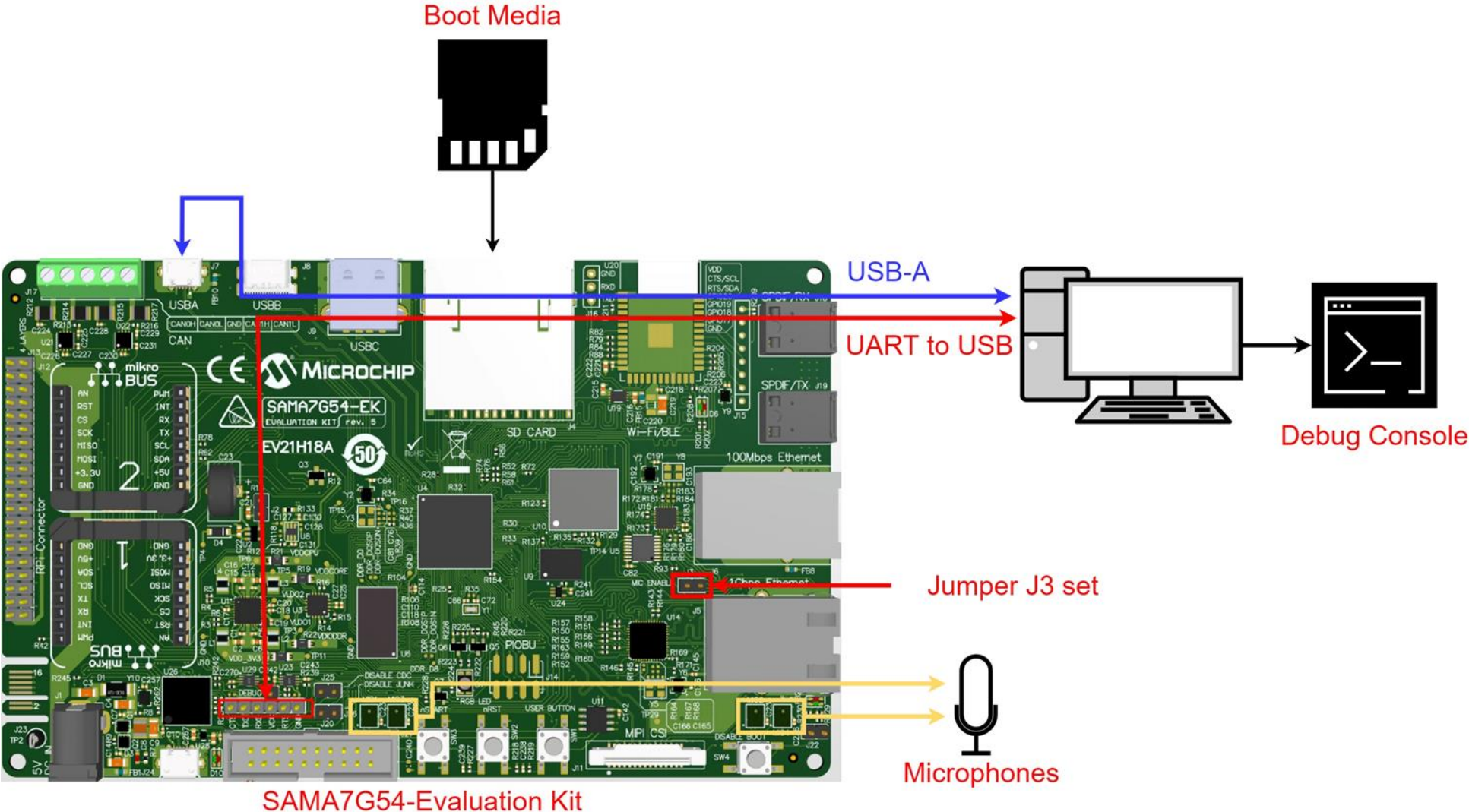
- **Convolutional Neural Network**

- Input shape of the model : 2D Data
- For audio signals : Fast Fourier Transform is applied to convert 1D audio signal to 2D data.



SAMA7 Demo Set-up and Output

Simple Audio Recognition



Running the application - Simple Audio Recognition

- The Application captures audio data from PDMs and saves it as .wav files.
- The processed audios are used as inputs for the models which returns outputs values
- Outputs values are indexed to the appropriate keyword and the probability that the audio match the keyword.
- Press the button and wait for the “Go !” to capture the sound and to process it through the model.

```
# python3 ./audio_reco_inference_button.py
*****
*** Welcome to the SAMA7G54-Ek Audio Recognition demo ***
*** Made with love by the MPU32 Marketing Team ***
*** Feel free to contact us if needed ***
*****
Starting Audio processing

Waiting for button press ....
User button press detected !
Ready ?
Go !
Recording done

Running inference...
Inference done in 227.53 ms
>>> Key Word Detected --> RIGHT
See all the outputs bellow :
Score for label down is 0.02 %
Score for label go is 0.00 %
Score for label left is 0.20 %
Score for label no is 0.00 %
Score for label right is 97.44 %
Score for label stop is 0.01 %
Score for label up is 0.01 %
Score for label yes is 2.32 %

Waiting for button press ....
```

Agenda

- Artificial Intelligence & Machine Learning
- AI Deploying Challenge on MPU
- TensorFlow & TensorFlow Lite
- AI Applications on MPU Using TensorFlow Lite Framework
- **Summary**

Summary

- With the popularity of IoT devices and embedded systems, it has become a demand and challenge to deploy artificial intelligence applications on microprocessors (MPUs) with low power consumption and limited computing power.
- Microchip's MPUs running TensorFlow Lite solutions will help you implement artificial intelligence embedded applications.

<https://www.microchip.com/en-us/solutions/machine-learning>



AI and Machine Learning

We offer solutions for a wide spectrum of users such as embedded systems engineers and data scientists. Learn how our simple ML design process can bring an ML engine to our MCUs, MPUs and FPGAs quickly and efficiently.

[Browse AI and Machine Learning](#)

Thank You
