Keyword Spotting on STM32

Group 4

Group Members

組長：VS6102124 高德龍

組員：N26101266 陳冠智

組員：NE6105012 許銘森

Outline

1. Motivation
2. Hardware Setup and System Framework
3. Probing PDM Microphone
4. MFCC and Neural Network
5. FreeRTOS Implementation
6. Performance Optimization
7. Experiment Results

1. Motivation

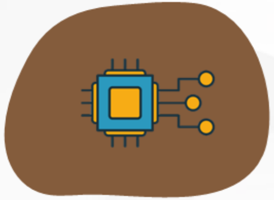
TinyML is a branch of machine learning and **embedded systems** research that looks into the types of models that can be run on small, low-power devices like microcontrollers. It delivers low-latency, low-power, and low-bandwidth model inference at edge devices.

Advantages:

- Eliminates the necessity of data transmission to a central server
- Fast inference with low latency
- Microcontrollers are Cheap, Prevalent and Low-power

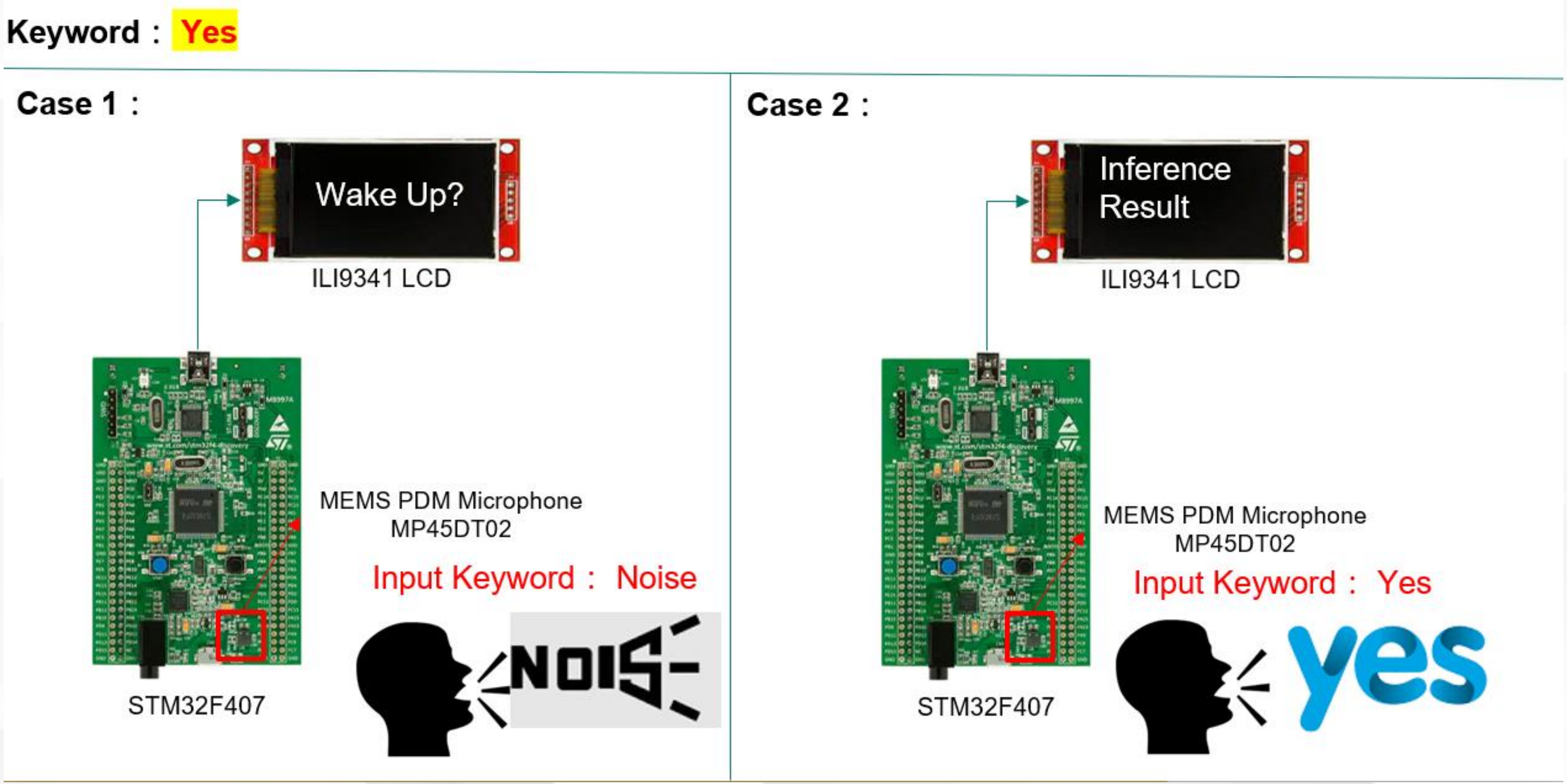
TinyML

FreeRTOS

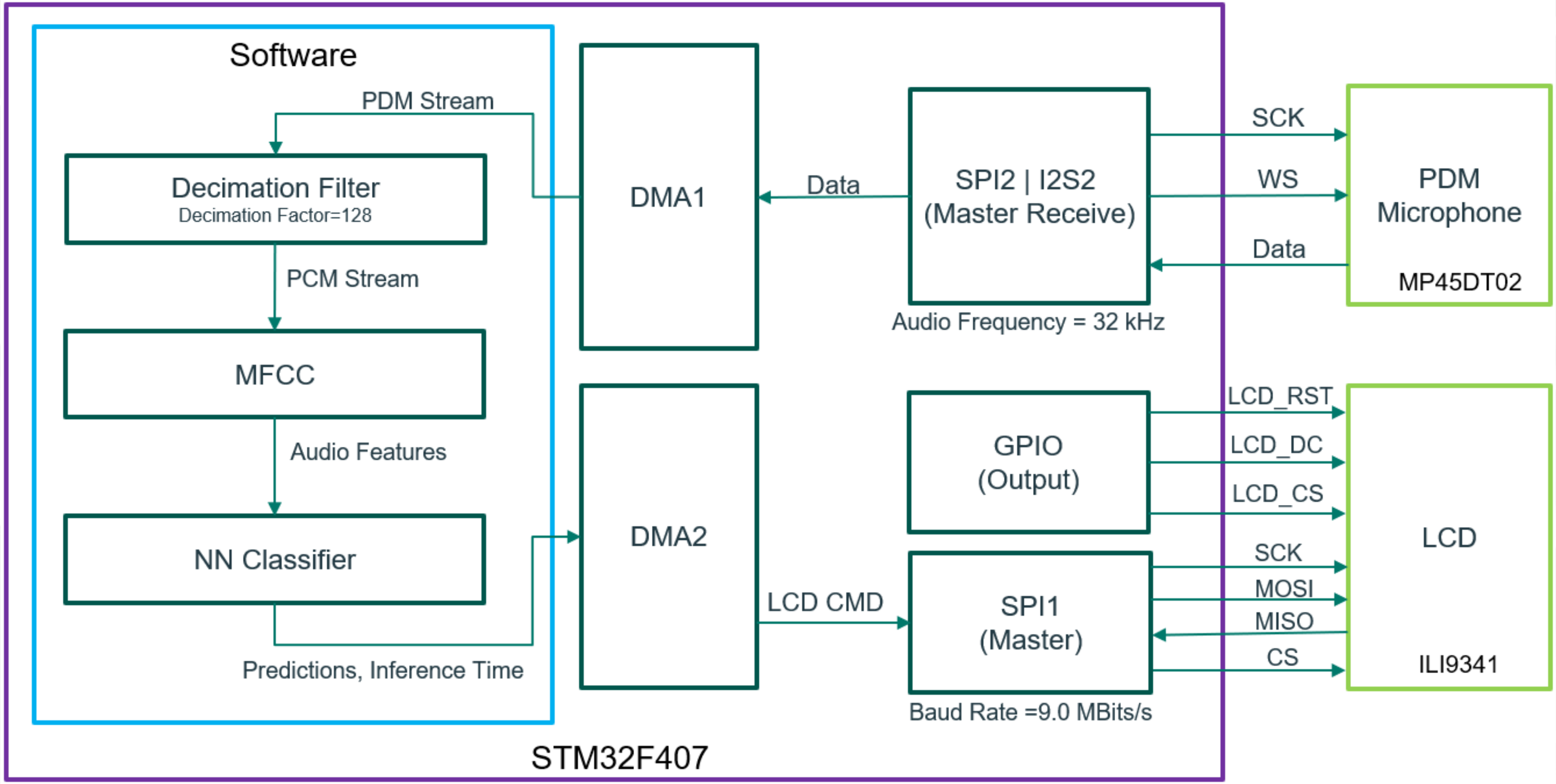


Embedded System

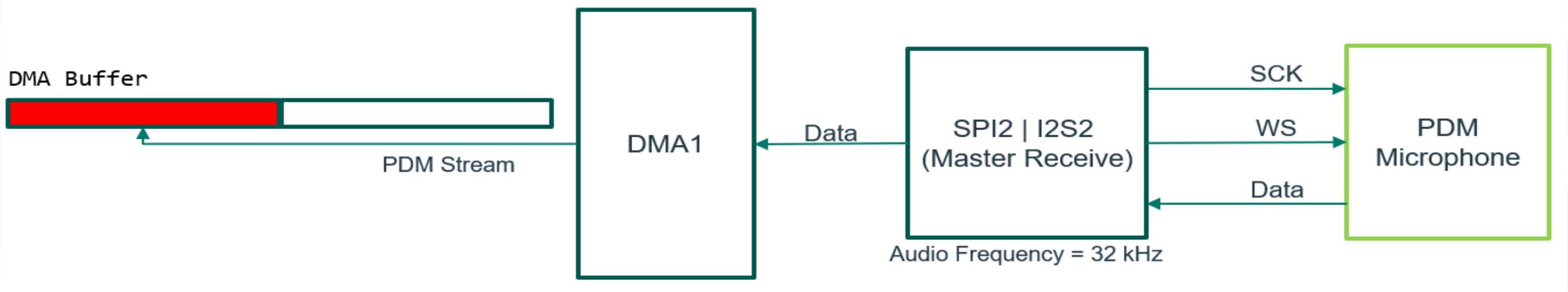
2.1 Hardware Setup and Function



2.2 System Framework

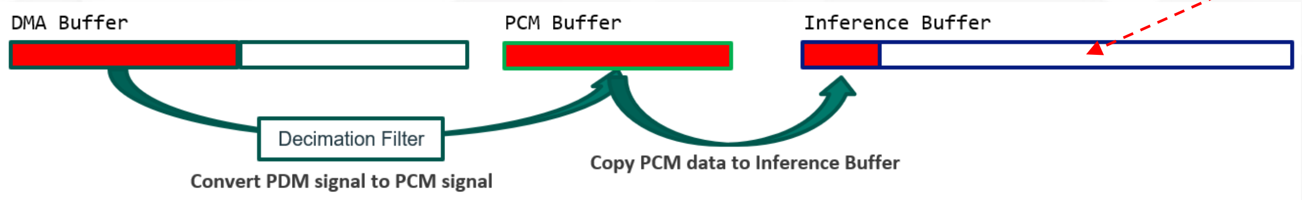


3. PDM Microphone



Callback when first or second half of DMA Buffer is full:

```
void HAL_I2S_RxHalfCpltCallback ()
void HAL_I2S_RxCpltCallback ()
```




```
typedef struct
{
  int16_t *buffers[2];
  uint8_t buf_select;
  volatile uint8_t buf_ready;
  volatile uint32_t buf_count;
  uint32_t n_samples;
} inference_t;
```

If Inference Buffer[0] is full, then it is ready for inference and switch to Inference Buffer[1] for receive

4.1 MFCC

PCM Signal

↓

1. Pre-emphasis

↓

2. Framing & Windowing

↓

3. Fast Fourier Transform

↓

4. Mel Filter Bank (Triangular Bandpass Filters)

↓

5. Log()

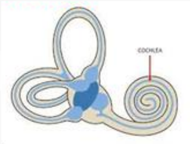
↓

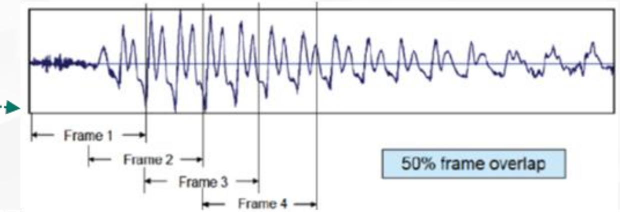
6. Discrete Cosine Transform (DCT)

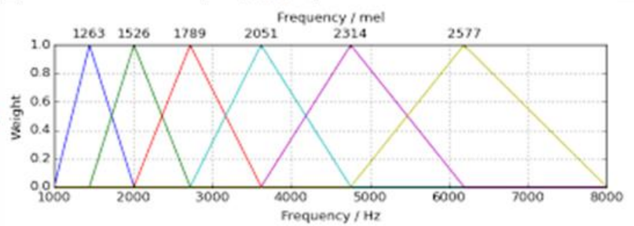
↓

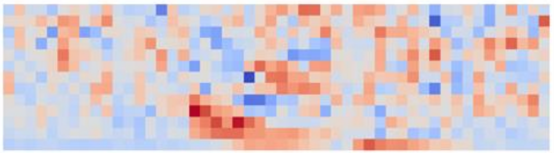
MFCC Features

The idea of MFCC is to convert time domain signals into frequency domain signal by mimicking cochlea function using Mel filters

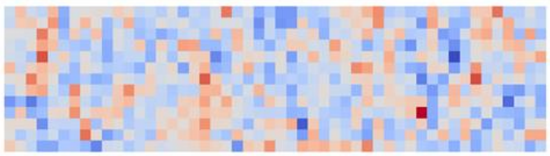






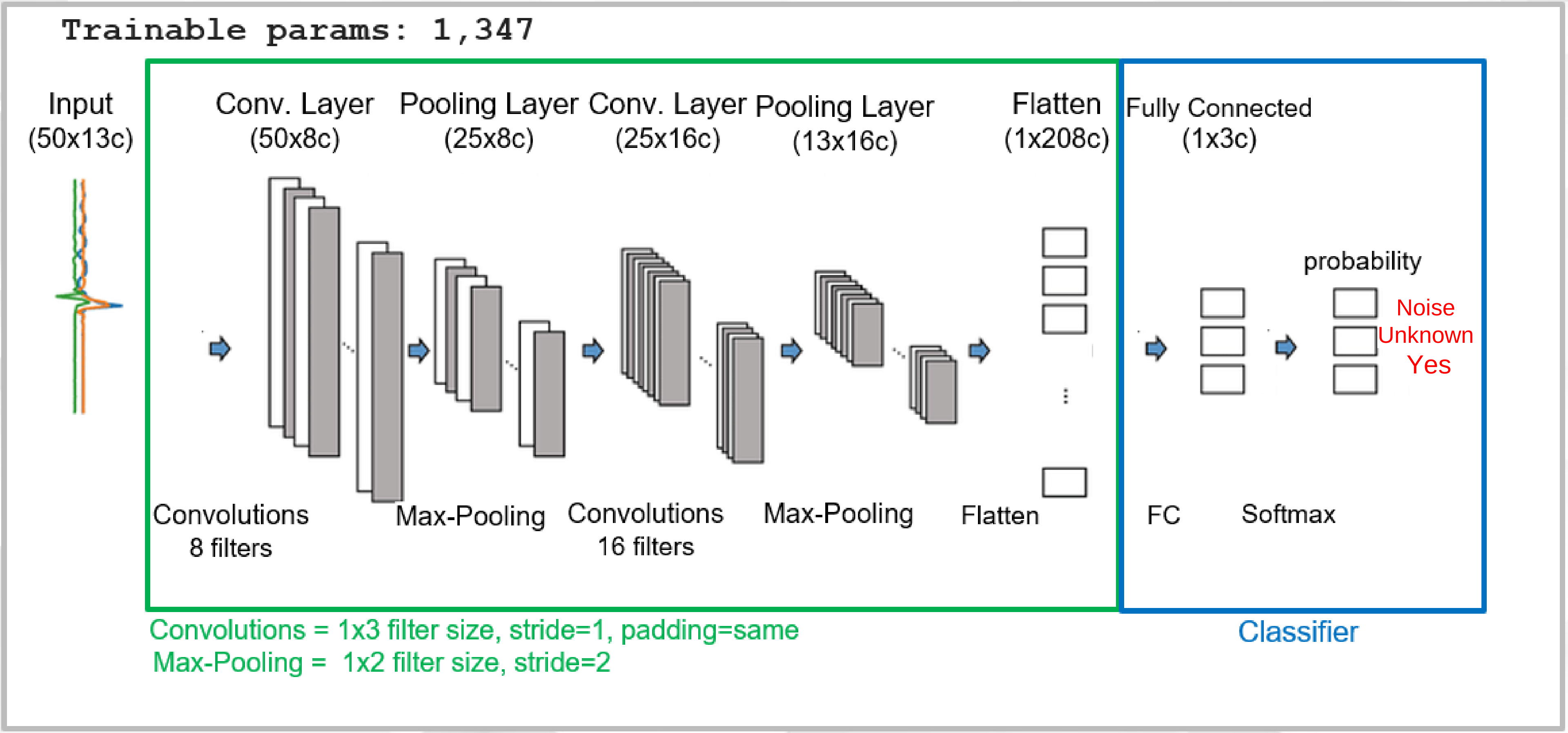


"Yes"

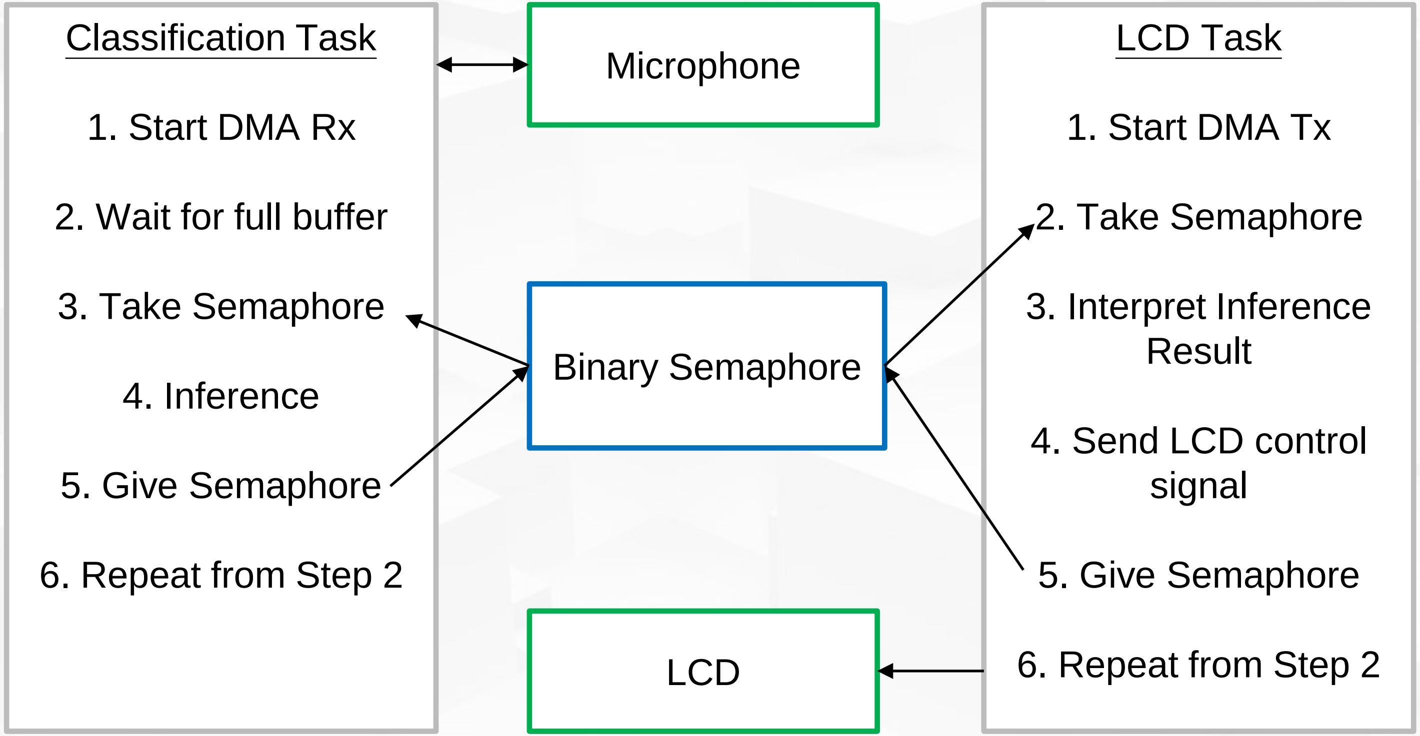


"Noise"

4.2 Convolutional Neural Network



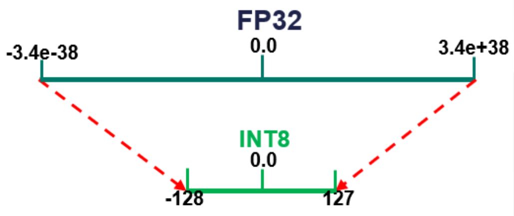
5. FreeRTOS Implementation



6. Performance Optimization - Quantization

1. Find out the largest and smallest value of FP32 parameters as $\underline{\mathrm{input}_{\mathrm{high}}}$ and $\underline{\mathrm{input}_{\mathrm{low}}}$
2. Mapping from FP32 space to INT8 space:

$$output = \frac{\lfloor (\mathbf{clamp}(\mathrm{input}, \mathrm{input}_{\mathrm{low}}, \mathrm{input}_{\mathrm{high}}) - input_{low}\ ) * S \rceil}{S} + input_{low}$$



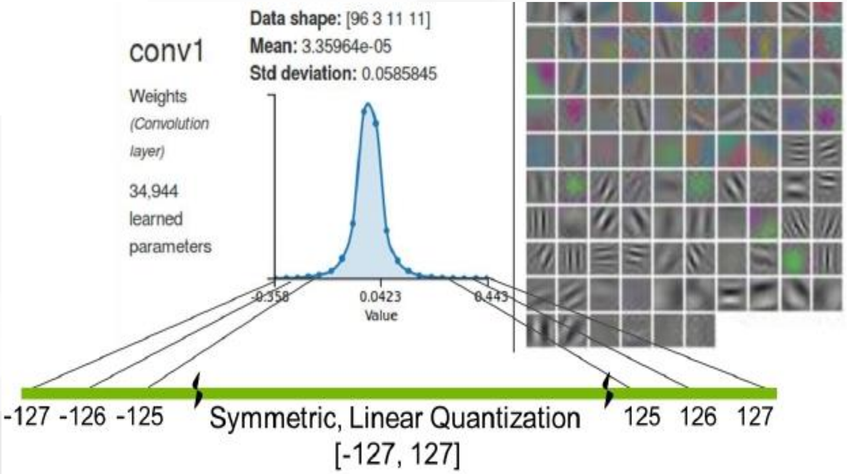
Restrict the input value range

➢ $\mathbf{clamp}(\mathrm{input}, \mathrm{input}_{\mathrm{low}}, \mathrm{input}_{\mathrm{high}}) = min\left(\max(input, input_{low})\, , input_{high}\right)$

Scaling Factor:

➢ $S = \frac{127-(-128)}{input_{high}-input_{low}} = \frac{255}{input_{high}-input_{low}}$

Pros:
Reduce model size
Reduce computational cost
Improve latency
Cons:
Accuracy may drop



7.1 Experimental Results – Neural Network

INT8 Model:

Accuracy = 93.8%   Loss = 0.18

	_NOISE	_UNKNOWN	YES
_NOISE	99.5%	0.5%	0%
_UNKNOWN	6.1%	89.4%	4.5%
YES	2.4%	4.7%	92.9%
F1 SCORE	0.95	0.92	0.94

FP32 Model:

Accuracy = 93.9%   Loss = 0.18

	_NOISE	_UNKNOWN	YES
_NOISE	99.5%	0.5%	0%
_UNKNOWN	5.3%	89.8%	4.9%
YES	2.0%	5.1%	92.9%
F1 SCORE	0.96	0.92	0.94

7.2 Experimental Results – DEMO

On-board Performance using INT8 model

MFCC Latency	120 ms
Neural Network Latency	10 ms
Peak RAM Usage (NN)	5.0K
Flash Usage (NN)	34.6K