# Défense Presentation

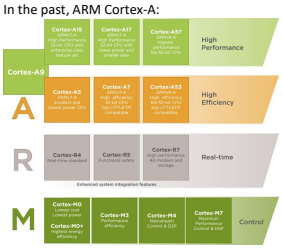
Sunday, 5 November, 2023 7:51 PM

## ARM Cortex-M

Original idea of the project (wearables):

- Proof that ARM Cortex-M chips can run AI

  In the past, ARM Cortex-A:



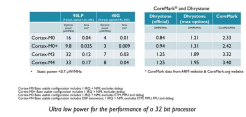

-

| | Application processors | Real-Time processors | Microcontroller processors |
|---|---|---|---|
| **Design** | High clock frequency, Long pipeline, High performance, Multimedia support (NEON instruction set extension) | High clock frequency, Long to medium pipeline length, Deterministic (low interrupt latency) | Short pipeline, ultra low power, Deterministic (low interrupt latency) |
| **System features** | Memory Management Unit (MMU), cache memory, TrustZone® security extension | Memory Protection Unit (MPU), cache memory, Tightly Coupled Memory (TCM) | Memory Protection Unit (MPU), Nested Vectored Interrupt Controller (NVIC), Wakeup Interrupt Controller (WIC) |
| **Targeted markets** | Mobile computing, smart phones, energy efficient servers, high end microprocessors | Industrial microcontrollers, automotives, Hard disk controllers, Baseband modem. | Microcontrollers, Deeply embedded systems (e.g. sensors, MEMS, mixed signal IC), Internet of Things (IoT) |

Table 1: Summary of processor characteristics

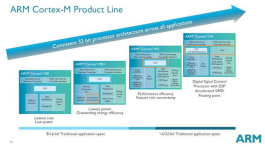
- Ultra-low power

  ARM Cortex-M Performance , Power and Area



Ultra low power for the performance of a 32 bit processor

- DSP

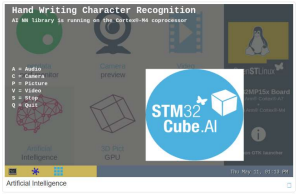



- https://www.linkedin.com/pulse/arm-cortex-m4-embedded-systems-exploring/

Demo:

https://wiki.st.com/stm32mpu/wiki/Getting_started/STM32MP1_boards/STM32MP157x-DK2/Let%27s_start/Use_the_demo_launcher

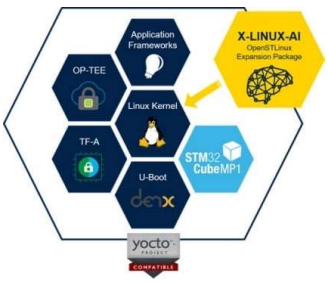
2.5 Artificial intelligence




This is a handwriting character recognition application:
1) It draws a character on the screen. The character is then analyzed by the firmware running on Arm® Cortex®-M4 side.
2) If the character is recognized, it is displayed on the left. Otherwise, a question mark is displayed.
3) When the recognized character is associated with an application, the application is launched:

- A: launches audio playback
- C: launches camera preview
- P: displays a picture
- V: launches a video playback
- S: stops the launched application
- Q: exits from the Artificial Intelligence demo.

STM MP1 Package + X-Linux-AI (ARM Cortex -M)

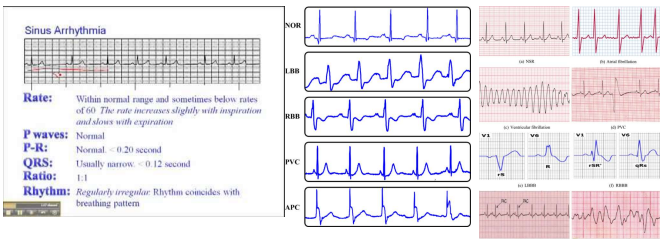




## Model for Heart Disease Prediction

Identification of Arrhythmia:

- Left Bundle Branch Block
- Normal
- Premature Atrial Contraction
- Premature Ventricular Contractions
- Right Bundle Branch Block
- Ventricular Fibrillation



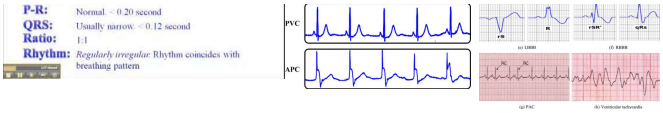

Dataset:

https://github.com/TriparnoChatterjee/Heart_Disease_Prediction_Using_ECG_Images

https://www.kaggle.com/datasets/erhmrai/ecg-image-data

https://www.kaggle.com/datasets/mohamedeldakroury8/ecg-heart-categorization-dataset-image-version

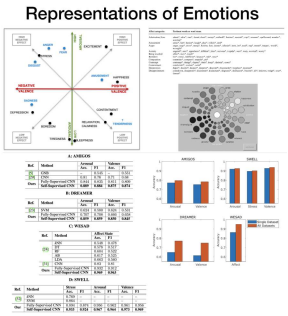
- Ventricular Fibrillation




Dataset:
https://github.com/TriparnoChatterjee/Heart_Disease_Prediction_Using_ECG_Images
https://www.kaggle.com/datasets/erhmrai/ecg-image-data
https://www.kaggle.com/datasets/mohamedeldakrory8/ecg-heart-categorization-dataset-image-version

## Model for Emotion Recognition

Type of Emotions:
- Discrete Emotional Model (DEM) -> happy, sad, angry, fear, etc.
- Affective Dimensional Model (ADM) -> valence, arousal, dominance
- Binary Emotional Model -> negative, positive




Dataset:
1. Amigos (http://www.eecs.qmul.ac.uk/mmv/datasets/amigos/download.html)
2. Dreamer (https://zenodo.org/records/546113#.YKLY0WYzaHs)
3. Wesad (https://archive.ics.uci.edu/dataset/465/wesad+wearable+stress+and+affect+detection)
4. Case (https://github.com/nosa999/Emotion-Recognition-from-ECG-Signal/blob/main/Case_ECG_Emotion
5. https://www.kaggle.com/code/danielfesalbon/ecg-signals-emotion-recognition (Model display low accura

Architecture:
CNN:
https://github.com/joergsimon/SSL-ECG-Paper-Reimplementaton/tree/main
- Pytorch
- include DEAP (EEG data)

## X-Linux-AI

Overview:
https://www.st.com/en/embedded-software/x-linux-ai.html

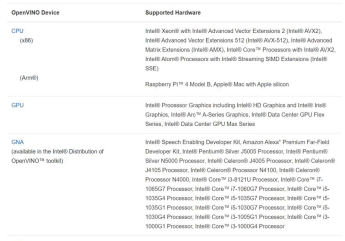
- **All-in-one AI solutions** for the entire STM32MPU serie
- **Pre-integrated** into Linux distribution based on ST environment
- Include **AI frameworks** to execute Neural Network models
- Include **AI model benchmark** application tools for MPU
- **Easy** application **prototyping** using Python language and AI frameworks Python API
- C++ **API** for embedded high-performance applications
- Optimized **open-source solutions** provided with source codes that allow extensive **code reuse** and **time savings**

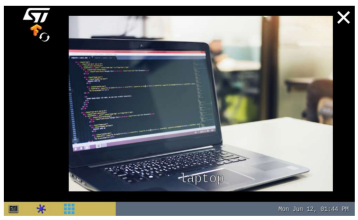
XNNPACK TensorFlow Lite - Leveraging CPU for ML inference:
https://www.st.com/en/embedded-software/x-linux-ai.html

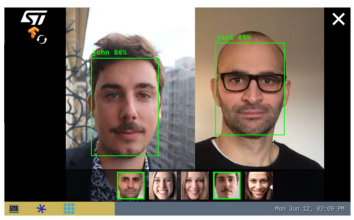
Package Library
https://wiki.st.com/stm32mpu/wiki/X-LINUX-AI_Starter_package

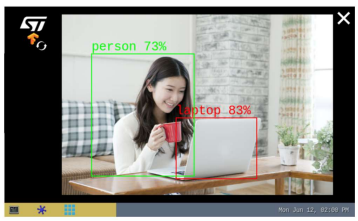
Application
https://wiki.st.com/stm32mpu/wiki/Category:AI_-_Application_examples
https://wiki.st.com/stm32mcu/wiki/AI:How_to_use_transfer_learning_to_perform_image_classification_on_STM32
https://wiki.st.com/stm32mcu/wiki/AI:How_to_use_Teachable_Machine_to_create_an_image_classification_application_on_STM32

Initial Idea: Intel OpenVINO
(https://docs.openvino.ai/2023.1/openvino_docs_OV_UG_supported_plugins_Supported_Devices.html)

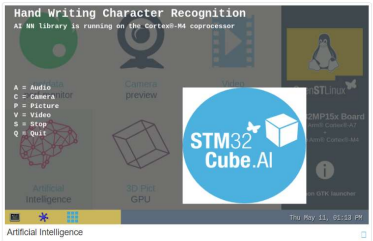






Artificial Intelligence

## Cloud Deployment (Optional)

Examples:
https://github.com/avnet-iotconnect/iotc-python-examples/tree/main/STM32MP157F-DK2_Demo
https://docs.iotconnect.io/iotconnect/git-resource/avnet-iotconnect-repository/
https://github.com/STMicroelectronics/meta-predmnt

https://www.st.com/resource/en/product_presentation/dsh-predmnt_getting_started.pdf

Running on:
AWS