Presented by:

Martin Croome
VP Business Development
GreenWaves Technologies

AT THE VERY EDGE

Enabling AI

ONNX
We are specialists in DSP / AI processing in devices with extreme energy constraints

MARKET

Wearables / Hearables

Smart home

Biosensing

IoT

TECHNICAL

Sensors

Visible Image

Sound

IR Image

Radar

Bio-sensors, ...

Acquisition
Inference
Output

100s µW to mWs in average operation
Few 10s mW for tens of GOPS
µWs in sleep mode

Communications

LoRa, BLE, Sigfox, NB-IoT, etc.

Our first product, GAP8, is in production with multiple design wins
Managing data movement - GAP AutoTiler

- GAP is not equipped with data caches
  - Silicon area
  - More important energy efficiency mostly due to hit ratio
- We can turn this weakness into an (energy) benefit if we can automate data transfers
- In practice a vast majority of traffic is predictable

Automatic data tiling and pipelined memory transfer interleaved with parallel call to compute kernel is solved by our “Autotiler” tool.
Complete GAPFlow

GAP code executed on GAP or SoC simulator
Experience with ONNX

• The good
  • Understandable operator set and structure
  • **GREAT** documentation
  • **GREAT** operator versioning system

• To be improved
  • Quantization
  • Fusion friendliness
Quantization

• Currently: Mix of Fake Quantization operators and a few quantized operators
  • What is the goal?
    • Express a quantized ONNX graph directly?
      • Are you going to provide quantized operators for every scheme with every different quantization technique?
    • Provide the information necessary to quantize a graph?
  • For the latter more is needed
    • Parameter statistics are easy
    • Activation statistics are not - best place to get them is training environment - all training data has been run forward through the graph
    • Absolutely necessary
      • Min/Max/Std/Mean
    • Nice to have
      • By channel
      • Outlier statistics
  • Proposal: Add statistics metadata to every (non-constant but why not all) tensor
Fusion friendliness

- Currently:
  - Some fused operators (GRU support is highly appreciated - not in TFLite)
  - Move towards functions for composed operators

- The problem:
  - Decomposed operators are like moving from Cow -> Mincemeat. Reverse direction is impossible with full optimisation

- The solution:
  - Encourage/force exporters to wrap the native high level operators that they are exporting in an ONNX function (a function is just a subgraph). If the function maps to an operator could this be set as well?
  - The namespace and function name should match the exporting platform function.
  - Reading the ONNX file you can either select an optimised version of the function (fusion) or run the operators in the subgraph
Thank You