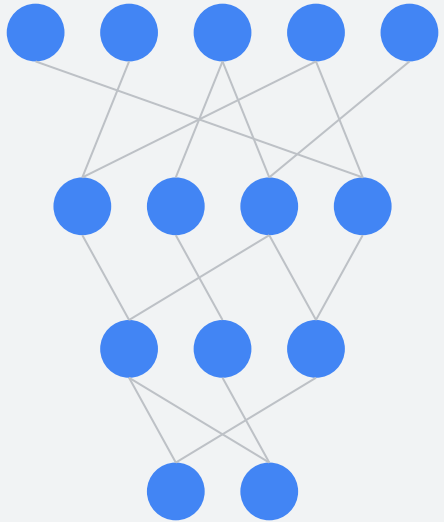
TFLite Micro: Model Format

The FlatBuffer File Format

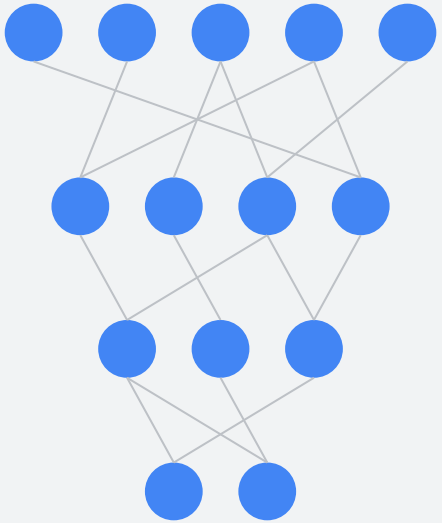
```
// Map the model into a usable data structure.  This doesn't involve any
// copying or parsing, it's a very lightweight operation.

    model = tflite::GetModel(g_model);
    if (model->version() != TFLIGHT_SCHEMA_VERSION) {
        TF_LITE_REPORT_ERROR(error_reporter,
                             "Model provided is schema version %d not equal "
                             "to supported version %d.",
                             model->versison(), TFLITE_SCHEMA_VERSION);

        return;
    }
```

```
// Map the model into a usable data structure.  This doesn't involve any
// copying or parsing, it's a very lightweight operation.

    model = tflite::GetModel(g_model);
    if (model->version() != TFLIGHT_SCHEMA_VERSION) {
        TF_LITE_REPORT_ERROR(error_reporter,
                             "Model provided is schema version %d not equal "
                             "to supported version %d.",
                             model->versison(), TFLITE_SCHEMA_VERSION);

        return;
    }
```

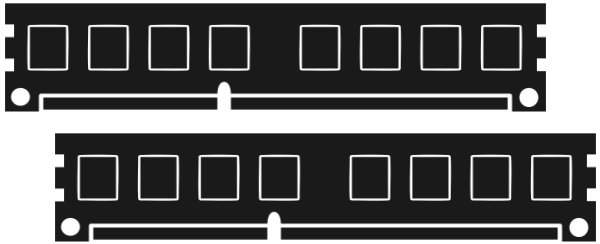
```
// Map the model into a usable data structure.  This doesn't involve
any
// copying or parsing, it's a very lightweight operation.

model = tflite::GetModel(g_model);
if (model->version() != TFLIGHT_SCHEMA_VERSION) {
    TF_LITE_REPORT_ERROR(error_reporter,
                         "Model provided is schema version %d not equal
"
                         "to supported version %d.",
                         model->versison(), TFLITE_SCHEMA_VERSION);

    return;
}
```



How is `g_model` stored?

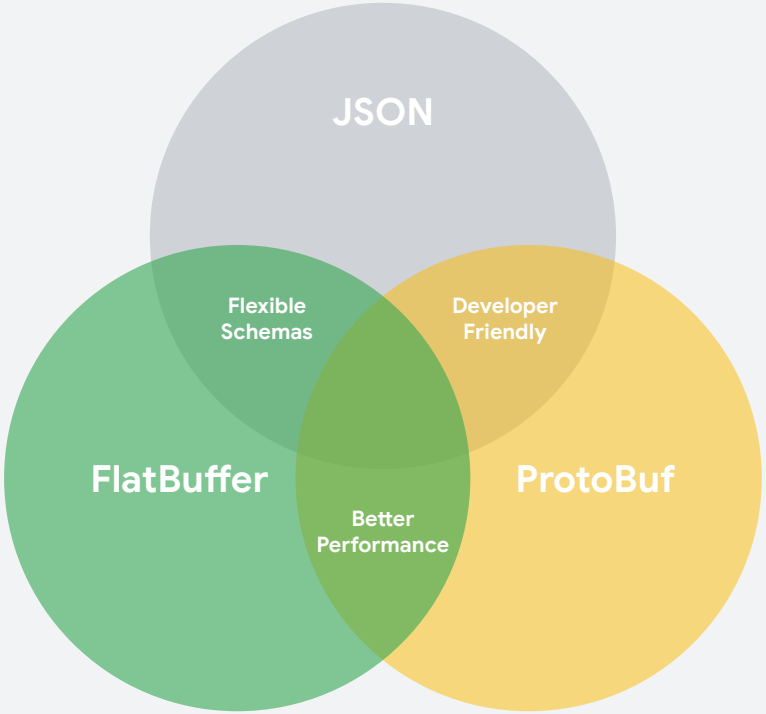


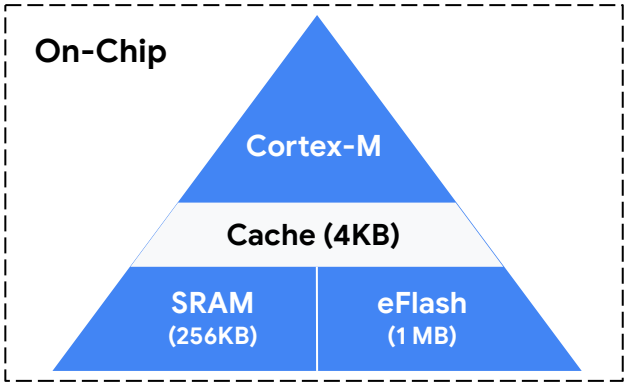
Serialization

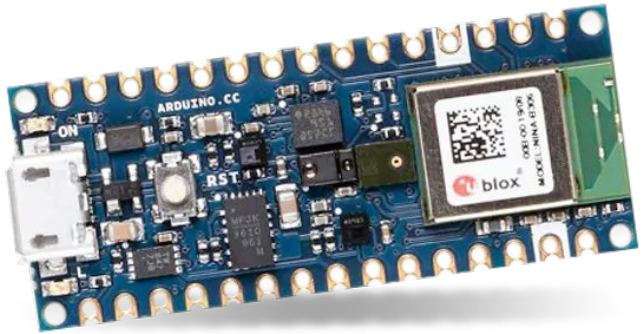


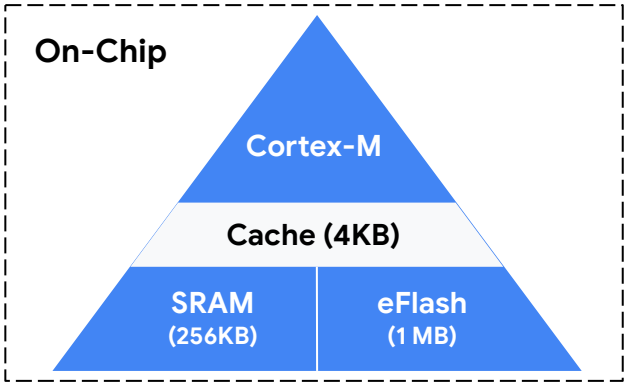
Serialization Libraries

- JSON
- ProtoBuf
- FlatBuffer



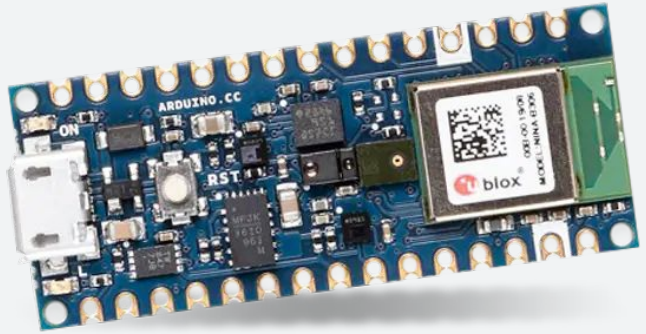

On-Chip
Cortex-M
Cache (4KB)
SRAM
(256KB)
eFlash
(1 MB)


ARDUINO.CC
ON
RST
blox



Hardware & Software Limitations

- Limited **OS support**
- Limited **compute**
- Limited **memory**



What is `g_model`?

- **Array of bytes**, and acts as the equivalent of a file on disk

- Holds **all** of the **information about the model**, its **operators**, their **connections**, and the trained **weights**

FlatBuffers

- Does **not require copies** to be made before using the data inside the model

FlatBuffers

- Does **not require copies** to be made before using the data inside the model
- The **format** is formally specified as a **schema file**

FlatBuffers

- Does **not require copies** to be made before using the data inside the model
- The **format** is formally specified as a **schema file**
- Schema file is used to automatically **generate code** to access the information in the **model byte array**

g_model FlatBuffer **Format**

Metadata (version, quantization ranges, etc)

Name	*Args*	*Input*	*Output*	*Weights*
Conv2D	3x3	0	1	2
FC	-	1	3	4
Softmax	-	3	5	-

Weight Buffers

Index	*Type*	*Values*
2	Float	0.01, 7.45, 9.23, ...
4	Int8	34, 19, 243, ...
...	...	...

How to **inspect** and **change** a model?

There's a **Python interface** to FlatBuffers, so you can manipulate a model file...

```
model = load_model_from_file('model.tflite')
for buffer in model.buffers:
    if buffer.data is not None and len(buffer.data) > 1024:
        original_weights = np.frombuffer(buffer.data, dtype=np.float32)
        munged_weights = np.round(original_weights * (1/0.02)) * 0.02
    save_model_to_file(model, 'model_modified.tflite')
```