

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->version(), TFLITE_SCHEMA_VERSION);

    return;
}
```

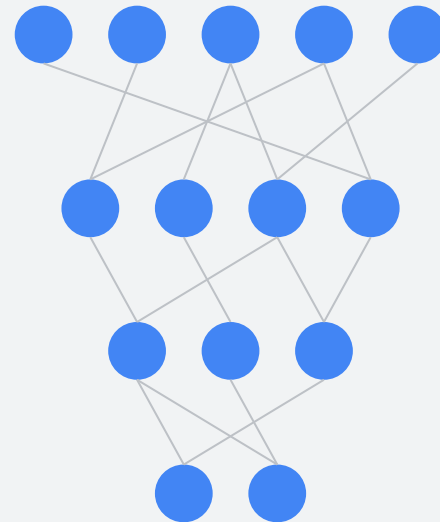
```
// Map the model into a usable data structure. This doesn't involve any
// copying or parsing, it's a very lightweight operation.

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->version(), TFLITE_SCHEMA_VERSION);

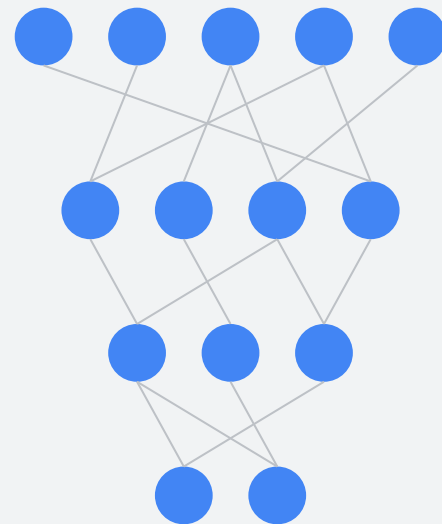
    return;
}
```

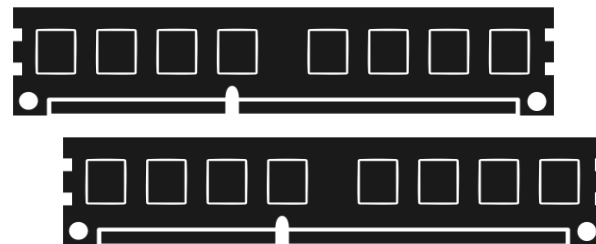
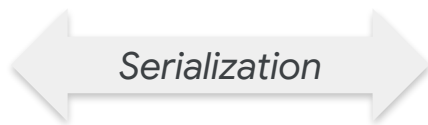
```
// Map the model into a usable data structure. This doesn't involve
any
// copying or parsing, it's a very lightweight operation.

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->version(), TFLITE_SCHEMA_VERSION);
    return;
}
```



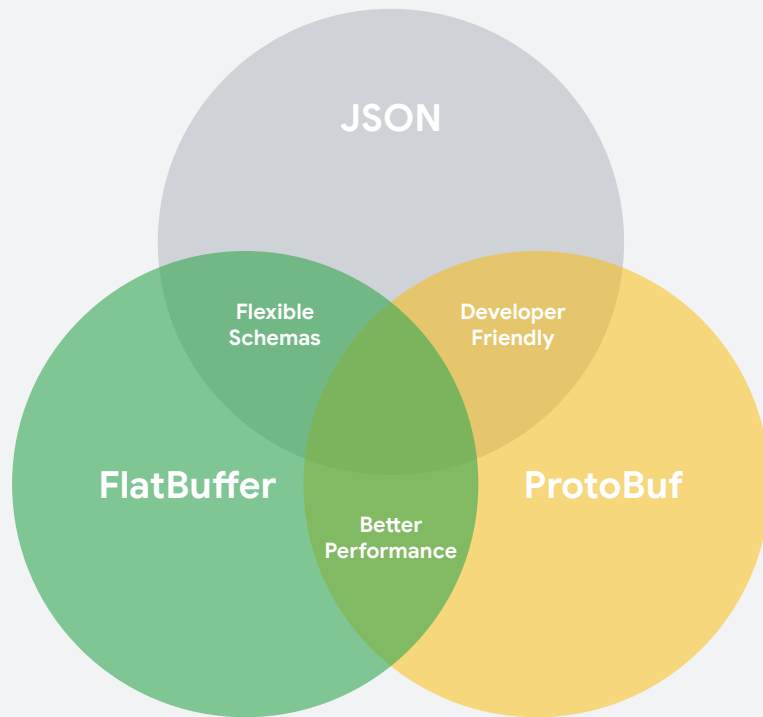
How is `g_model` stored?

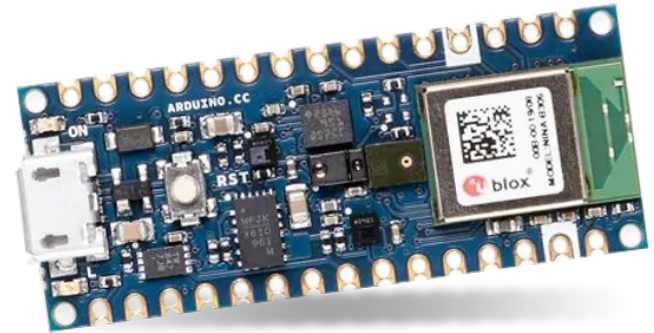
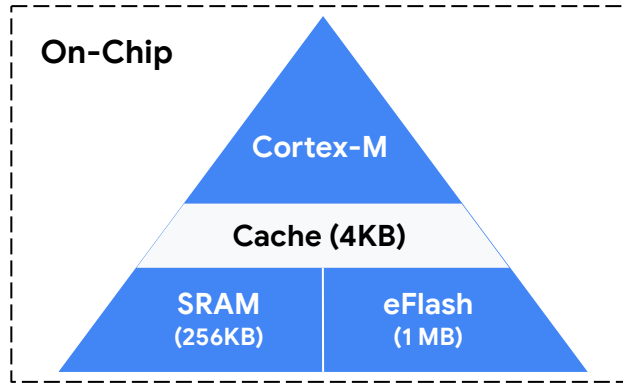


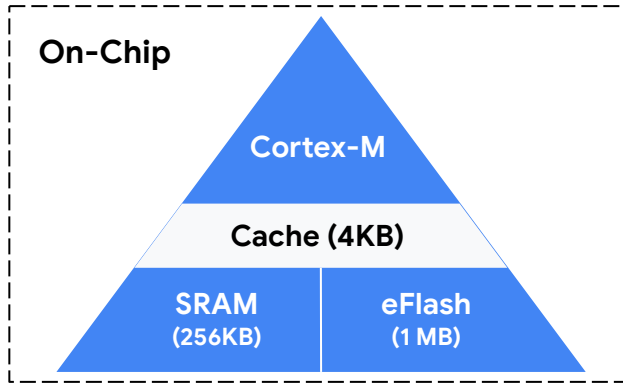


Serialization Libraries

- JSON
- ProtoBuf
- FlatBuffer

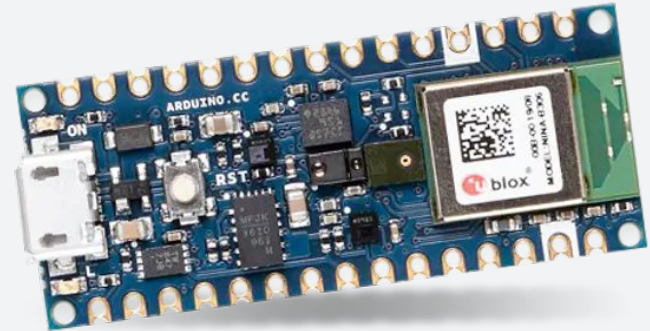






Hardware & Software Limitations

- 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**

```
5 // Automatically created from a TensorFlow Lite flatbuffer using the command:
17 // xxd -i model.tflite > model.cc
18
19 // This is a standard TensorFlow Lite model file that has been converted into a
20 // C data array, so it can be easily compiled into a binary for devices that
21 // don't have a file system.
22
23 // See train/README.md for a full description of the creation process.
24
25 #include "model.h"
26
27 // Keep model aligned to 8 bytes to guarantee aligned 64-bit accesses.
28 alignas(8) const unsigned char g_model[] = {
29     0x1c, 0x00, 0x00, 0x00, 0x54, 0x46, 0x4c, 0x33, 0x00, 0x00, 0x12, 0x00,
30     0x1c, 0x00, 0x04, 0x00, 0x08, 0x00, 0x0c, 0x00, 0x10, 0x00, 0x14, 0x00,
31     0x00, 0x00, 0x18, 0x00, 0x12, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00,
32     0x60, 0x09, 0x00, 0x00, 0xa8, 0x02, 0x00, 0x00, 0x90, 0x02, 0x00, 0x00,
33     0x3c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
34     0x0c, 0x00, 0x00, 0x00, 0x08, 0x00, 0x0c, 0x00, 0x04, 0x00, 0x08, 0x00,
35     0x08, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x0b, 0x00, 0x00, 0x00,
36     0x13, 0x00, 0x00, 0x00, 0x6d, 0x69, 0x6e, 0x5f, 0x72, 0x75, 0x6e, 0x74,
37     0x69, 0x6d, 0x65, 0x5f, 0x76, 0x65, 0x72, 0x73, 0x69, 0x6f, 0x6e, 0x00,
38     0x0c, 0x00, 0x00, 0x00, 0x48, 0x02, 0x00, 0x00, 0x34, 0x02, 0x00, 0x00,
39     0x0c, 0x02, 0x00, 0x00, 0xfc, 0x00, 0x00, 0x00, 0xac, 0x00, 0x00, 0x00,
40     0x8c, 0x00, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x34, 0x00, 0x00, 0x00,
41     0x2c, 0x00, 0x00, 0x00, 0x24, 0x00, 0x00, 0x00, 0x1c, 0x00, 0x00, 0x00,
42     0x04, 0x00, 0x00, 0x00, 0xfe, 0xfd, 0xff, 0x04, 0x00, 0x00, 0x00,
43     0x05, 0x00, 0x00, 0x00, 0x31, 0x2e, 0x35, 0x2e, 0x30, 0x00, 0x00, 0x00,
44     0x7c, 0xfd, 0xff, 0xff, 0x80, 0xfd, 0xff, 0xff, 0x84, 0xfd, 0xff, 0xff,
45     ...,
46     0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x00, 0x06, 0x00, 0x05, 0x00,
47     0x06, 0x00, 0x00, 0x00, 0x00, 0x72, 0x0a, 0x00, 0x0c, 0x00, 0x07, 0x00,
48     0x00, 0x00, 0x08, 0x00, 0x0a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x09,
49     0x04, 0x00, 0x00, 0x00};
50 const int g_model_len = 2512;
```

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')
```