Modeling Power in gem5

What we will cover

- Idea behind power modeling
- The `MathExprPowerModel`
- An example

Power modeling

- gem5 supports "activation count" based power models
- This is kind of like McPAT/Wattch/etc. but it can be done from gem5
- The correct constants are difficult to find
  - CACTI is one possibility, but...

For instance

$$P = \frac{N_{cache\_accesses} * 18\mu J + N_{cache\_misses} * 1\mu J}{s}$$

> We are building *power* models, not energy models. So, don't forget to convert to *Watts* not *Joules*.

Power models in gem5

gem5 has a generic power model that exposes `getDynamicPower` and `getStaticPower` to python

Each SimObject can have different power models for different power states (e.g., on, off, clock gated, and SRAM retention)

There is also a thermal model which uses a RC circuit model.
We won't talk about the thermal model today.

See `gem5/src/sim/power/`

We will use `MathExprPowerModel`, but you can also make your own models.
There should be a new `PythonFunction` power model coming soon.

MathExprPowerModel

We will be using the `MathExprPowerModel`
See `gem5/src/sim/power/MathExprPowerModel.py`

This let's you specify a math expression like we saw previously for the power model.

You can use statistics, voltage, and information from your thermal model.

The voltage comes from the voltage domain for the object (specified in the `ClockedObject`).

Some caveats before we get into the example

We (currently) don't provide any constants.

We're going to see some hacks because the stdlib doesn't support power models.

This can't be emphasized enough: Be careful when using power models!

L3 cache power model

Let's start with the L3 cache that we made previously using the classic caches.

Here's some code that we provided

```python
from m5.objects import PowerModel, MathExprPowerModel

class L3PowerModel(PowerModel):
    def __init__(self, l3_path, **kwargs):
        super().__init__(**kwargs)
        # Choose a power model for every power state
        self.pm = [
            L3PowerOn(l3_path),  # ON
            L3PowerOff(),  # CLK_GATED
            L3PowerOff(),  # SRAM_RETENTION
            L3PowerOff(),  # OFF
        ]
```

Add the power model for the "on" state

```
class L3PowerOn(MathExprPowerModel):
    def __init__(self, l3_path, **kwargs):
        super().__init__(**kwargs)
        self.dyn = f"({l3_path}.overallAccesses * 0.000_018_000 \
                 + {l3_path}.overallMisses * 0.000_001_000)/simSeconds"
        self.st = "(voltage * 3)/10"
```

This power model uses the equation from the earlier slide.

We get the "overallAccesses" stat and the "overallMisses" stat from the cache.

You can look at the stats.txt file for all of the stats you can use.

We also divide by the number of seconds simulated to calculate the *watts* instead of *joules*. (If you want to get a delta time... good luck)

Adding the power model

Add the following code to your `cache_hierarchy`.

```
    def add_power_model(self):
        self.l3_cache.power_state.default_state = "ON"
        self.l3_cache.power_model = L3PowerModel(self.l3_cache.path())
```

> This needs to be called after `board._pre_instantiate` but before `m5.instantiate`. This is currently not supported in the standard library.

You need to get the "Path" to the L3 cache so that you can get the stat.
The path is the "full name" for the object.
The `SimObject` has a function to get the path, but it's only valid after creating the `Root` object.

Run the code

```
gem5 test-cache.py
```

See the output from `stats.txt`

```
grep power_model m5out/stats.txt
```

```
board.cache_hierarchy.l3_cache.power_model.dynamicPower  2011.504302
board.cache_hierarchy.l3_cache.power_model.staticPower     0.300000
board.cache_hierarchy.l3_cache.power_model.pm0.dynamicPower  2011.504302
board.cache_hierarchy.l3_cache.power_model.pm0.staticPower     0.300000
board.cache_hierarchy.l3_cache.power_model.pm1.dynamicPower           0
board.cache_hierarchy.l3_cache.power_model.pm1.staticPower           0
```

2 KW for the cache????? that seems wrong...