Zephyr® Project
Developer Summit 2022

SMP support for RISC-V* The road so far and what lies ahead

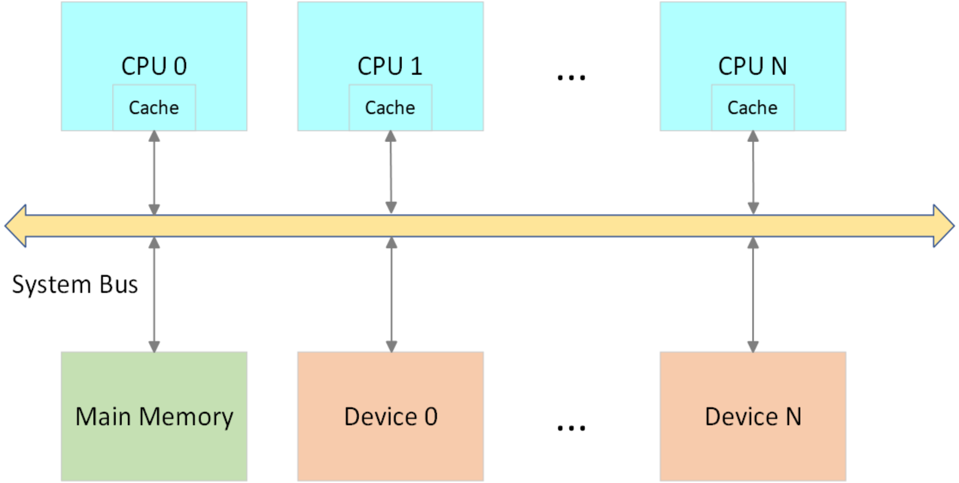
Ederson de Souza, Intel Corporation

Agenda

- SMP?
- Community interest
- Getting in
    - And almost out
- Next steps
    - Real platforms ***not*** quite SMP
    - Possible approaches

SMP?

- Symmetric Multiprocessing
- Identical processors
- Share memory and I/O access



Community interest

- First pull requests from end of 2020
  - By @katsuster, #29105
  - By @cwshu (Andes Technology), #29131
  - Unfortunately, efforts stalled
    - Lack of review; maintainers not involved;
- Some out-of-tree work announced on the mailing list by the end of 2021
  - Support for specific platform - Microchip PolarFire* SOC
- Clear interest, not so clear path forward

Getting in

- Another PR by the beginning of 2022
  - Just a bit more than one year from first one
- Isn't it just a matter of implementing `arch_switch()`
  - Well yes, but actually no
- Finally merged
  - It pays to be a bit pushier to the maintainers

And then it was refactored out…

…but in again

Community interest (II)

- RISC-V asm was quite messy
  - SMP patches improved things a little, with overall code reduction
- Refactor dialed up to 11
  - Thoroughly changed – and reduced – asm code
  - Some SMP patches were "too painful" to maintain
  - But features were nicely added back
  - Available on v3.1.0

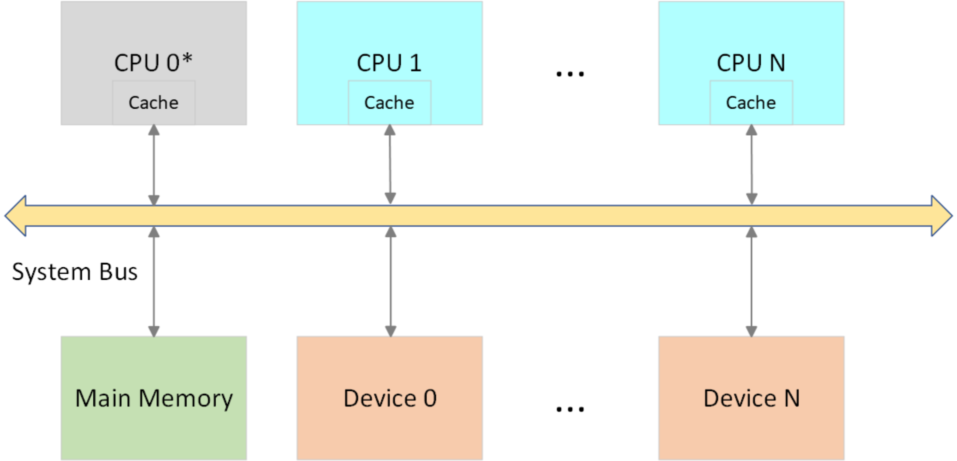
Next steps

- Current support is for Qemu only
  - Good for development, not so good for The Real World™
- Support for real platforms
  - Simulators, like Renode*, can be really handy
  - But there's a catch...

Real platforms *not* quite SMP

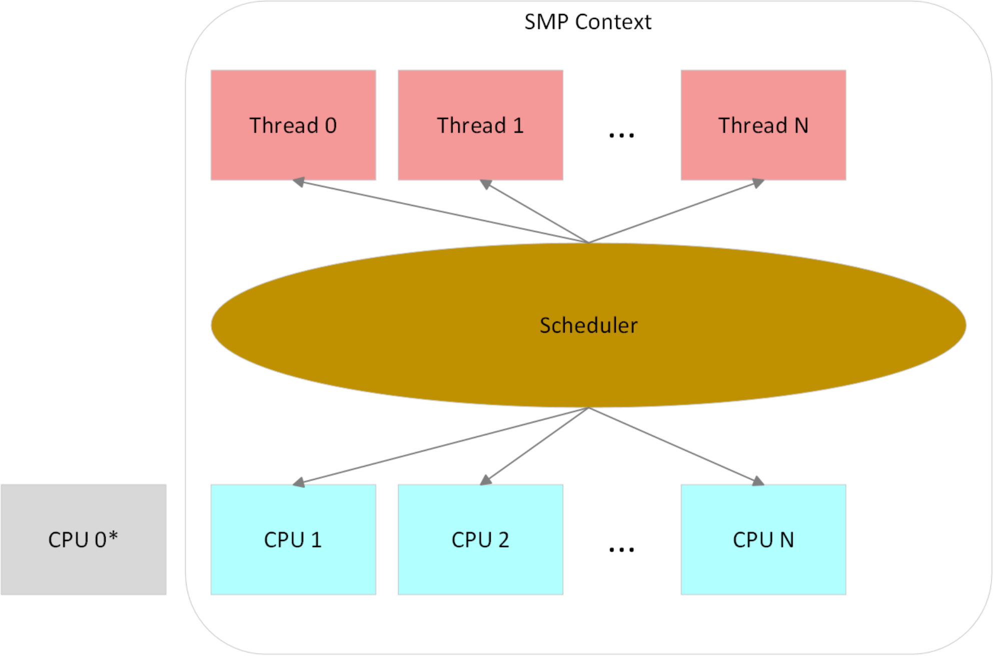
Some examples

- Microchip PolarFire SOC
- SiFive* U-54/U-74
- Usually:
  - 1 RV64IMAC (Monitor)
    - Maybe not even LR/SC
  - 4 RV64GC(B) (Application)
- Not quite SMP, but not that far too
  - Same memory, I/O



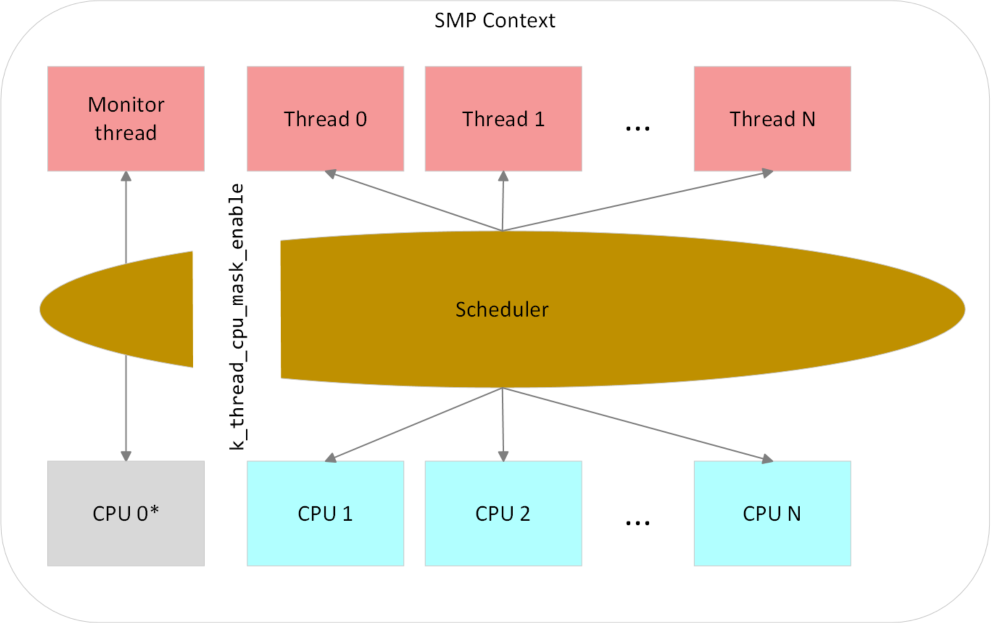
Approaches (I)

- Ignore the Monitor core
  - May still have to manage interrupts on it
- Better to have 4 cores than 1
  - But losing one core may be undesirable
  - Maybe load another Zephyr in the first core?
- Something else?



Approaches (II)

- Just pretend everything is SMP
  - But Zephyr only manages up to 4 processors
- Use `k_thread_cpu_mask_enable()` to restrict SMP threads to symmetric cores
  - Maybe `k_busy_wait()` on Monitor core to avoid it ever trying to participate on scheduling?
  - What about interrupts?
- Something else?



Approaches (III)

- Make Zephyr scheduler aware of quasi-SMP scenarios
- What about interrupts?
- What about missing atomic instructions?
- Something else?

Questions?

intel®

Zephyr® Project
Developer Summit 2022

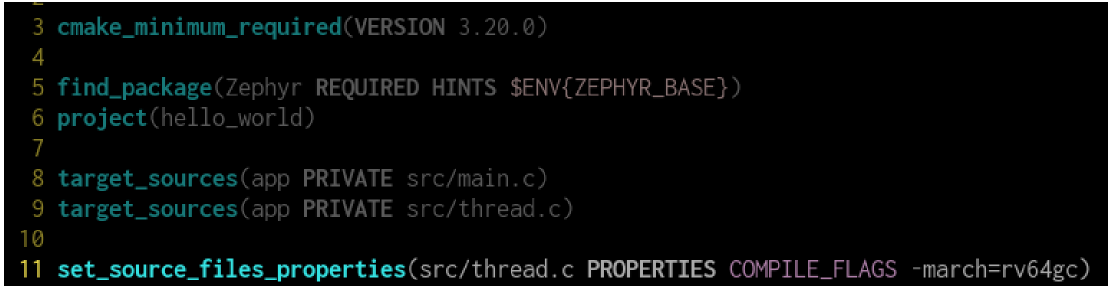
Thank you!

*Other names and brands may be claimed as the property of others.

Approaches (II) - Simulator test, adding FP numbers

- Three threads
  - Main runs on Monitor
  - Float ones run on two Application cores
- Two object files
  - One for Main
  - One for Float, with tweaked options

CMakeLists.txt



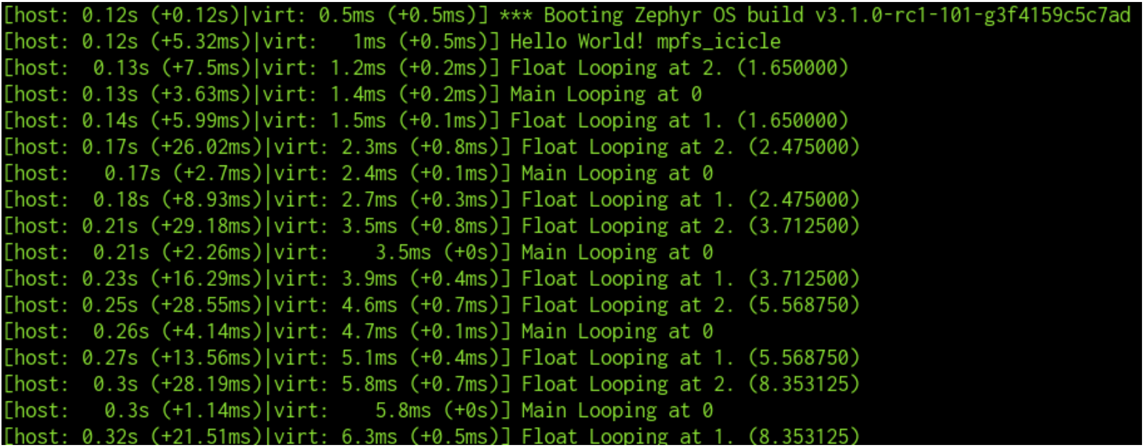
```
cmake_minimum_required(VERSION 3.20.0)

find_package(Zephyr REQUIRED HINTS $ENV{ZEPHYR_BASE})
project(hello_world)

target_sources(app PRIVATE src/main.c)
target_sources(app PRIVATE src/thread.c)

set_source_files_properties(src/thread.c PROPERTIES COMPILE_FLAGS -march=rv64gc)
```

Simulator run excerpt



```
[host: 0.12s (+0.12s)|virt: 0.5ms (+0.5ms)] *** Booting Zephyr OS build v3.1.0-rc1-101-g3f4159c5c7ad
[host: 0.12s (+5.32ms)|virt:   1ms (+0.5ms)] Hello World! mpfs_icicle
[host:  0.13s (+7.5ms)|virt: 1.2ms (+0.2ms)] Float Looping at 2. (1.650000)
[host: 0.13s (+3.63ms)|virt: 1.4ms (+0.2ms)] Main Looping at 0
[host: 0.14s (+5.99ms)|virt: 1.5ms (+0.1ms)] Float Looping at 1. (1.650000)
[host: 0.17s (+26.02ms)|virt: 2.3ms (+0.8ms)] Float Looping at 2. (2.475000)
[host:   0.17s (+2.7ms)|virt: 2.4ms (+0.1ms)] Main Looping at 0
[host:  0.18s (+8.93ms)|virt: 2.7ms (+0.3ms)] Float Looping at 1. (2.475000)
[host: 0.21s (+29.18ms)|virt: 3.5ms (+0.8ms)] Float Looping at 2. (3.712500)
[host:  0.21s (+2.26ms)|virt:    3.5ms (+0s)] Main Looping at 0
[host: 0.23s (+16.29ms)|virt: 3.9ms (+0.4ms)] Float Looping at 1. (3.712500)
[host: 0.25s (+28.55ms)|virt: 4.6ms (+0.7ms)] Float Looping at 2. (5.568750)
[host:  0.26s (+4.14ms)|virt: 4.7ms (+0.1ms)] Main Looping at 0
[host: 0.27s (+13.56ms)|virt: 5.1ms (+0.4ms)] Float Looping at 1. (5.568750)
[host:  0.3s (+28.19ms)|virt: 5.8ms (+0.7ms)] Float Looping at 2. (8.353125)
[host:   0.3s (+1.14ms)|virt:    5.8ms (+0s)] Main Looping at 0
[host: 0.32s (+21.51ms)|virt: 6.3ms (+0.5ms)] Float Looping at 1. (8.353125)
```