XinuPi3?

XinuPi3 is the first open source lightweight instructional operating system that uses multiple cores
- It is much easier and much more effective to teach using a real operating system rather than a simulation
- It is more important than ever to teach multi-core concepts to the younger generation,
  o Even smartphones run on multiple cores
- XinuPi3 can assist, not only in teaching these very important multi-core concepts, but also in teaching typical operating system concepts in a new way.

How it looks

Running a core is as simple as:

```
*(volatile fn *)(CORE_MBOX_BASE + CORE_MBOX_OFFSET * num) = Core1Setup;
```

This line of code instructs a core numbered 1 - 3 to execute at address Core1Setup

Once a core has been awoken,

```
ldr r0, =core_init_sp
ldr sp, [r0, #4]
```

The stack must be set up in ARM assembly code. This is the only requirement needed to start running code on a core, but if this requirement isn’t immediately met, the core will be stuck.

Using just 3 lines of code (in reality, only 2 of those three lines are needed), any student can run code on multiple cores. Cool!

References


It’s Easier

- Can teach concepts about processes without having to first teach how to switch between processes
- Easier to think about processes running concurrently
  o Helps students understand why mutexes are necessary and how they work

It’s Harder

- Mutexes and semaphores don’t work the same way
- Harder to control what is running and when
- Processes actually running at the same time, which is cool but can get messy

It’s Worth It

- Old ways of using mutexes and semaphores will become unnecessary to know
- Students become more satisfied if knowing they are working on new, unchartered territory
- Employers would rather hire employers with experience with multi-core systems

Special Thanks To:
- Phil Elwell for helping us get started
- David Welch for teaching about multi-cores
- Goswin von Brederlow for much-needed code

Acknowledgments

Special Thanks To:
- Phil Elwell for helping us get started
- David Welch for teaching about multi-cores
- Goswin von Brederlow for much-needed code