

Response to Programming Models

The Good

- Diagnostics and other “stuff” on the Stuff slide
 - Right on!
 - **Concern: Need more details on the actual API (how are you actually going to do it?).**
 - Users need to be able to use it.
 - Needs to be high enough priority to actually be implemented.

Good and Bad

- Memory sync and programmer access:
 - Bad: Long term we don't want to think about memory system implementation details.
 - Good: In the mean time (maybe forever?), control over this level of detail is necessary.
 - Some contention on this.

Good and Bad (cont)

- History: Pragmas/options don't always work.
- Programming features don't deliver promise:
 - Broken pragmas.
 - Ignored compiler options/pragmas.

Objectionable proposals

- None.
- At least at the level of detail presented.

Things you should have said

- Memory controller support for memory errors.
 - Example: data watch points.
- Support for graceful evolution.
 - Current investments are huge.
 - No ephemeral features (codes live for decades).