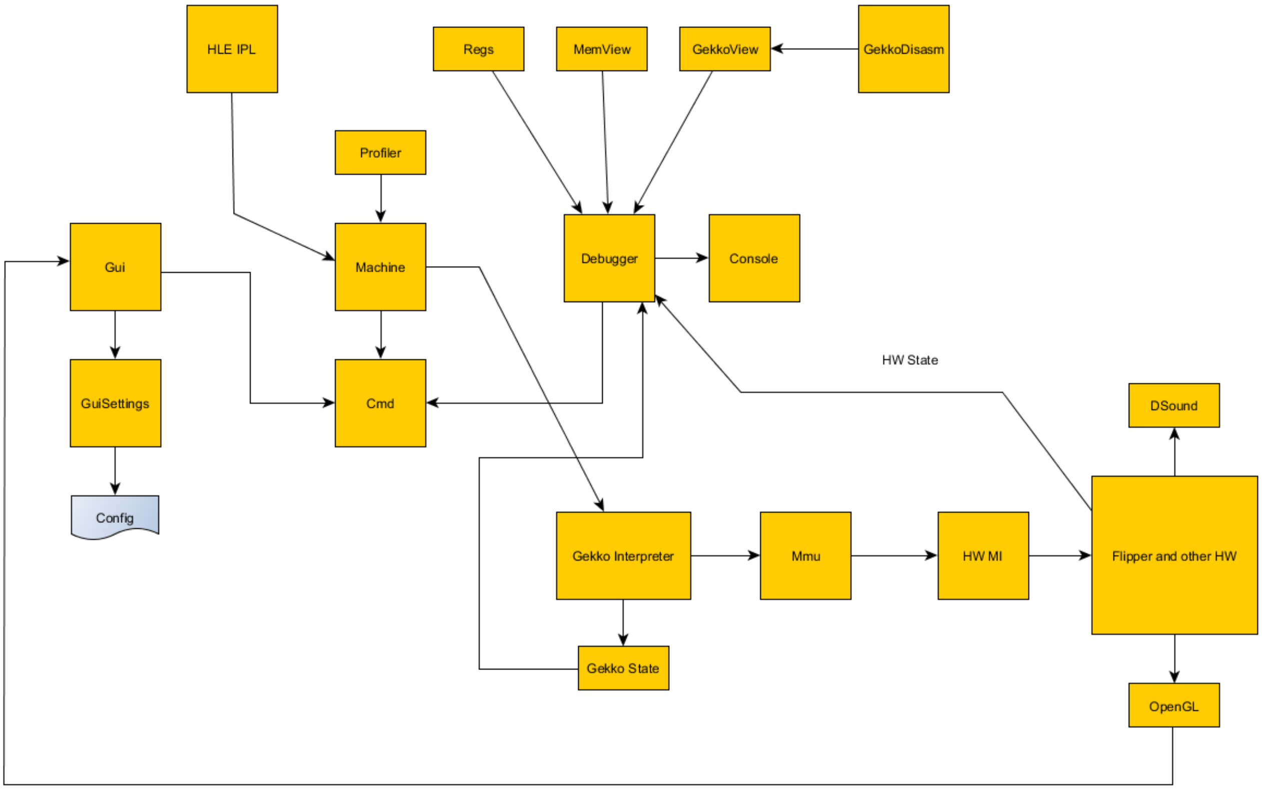
Dolwin Refactoring: Stage 1

This document describes ass movements to get rid of high-priority issues from the TODO.

Following diagram illustrates preliminary design of Dolwin components:



Steps:

1. Source code tree refactoring

Done. New code is placed in Source directory, while deprecated code still maintain in SRC directory.
New VisualStudio solution is located in DolwinNew.

2. Implement some Utils

- Listutils: Double-linked lists
- Cmd: command processor for debugger and other major parts. Now any subsystem can “register” command, which can be executed from anywhere.

- Config: File-based configs
- FileWrap: Portable file access
- Threads: Thread creation/sleep manager
- Spinlock: Thread synchronization spinlocks (based on win32 critical sections)
- GekkoDisasm: Gekko disassembler for debugger
- Machine: (deprecated Emulator.cpp). General GC control hub, through debug commands. Also contain initialization / shutdown code.

3. Write some basic Gui to test Machine runflow.

4. Implement debugger core Console.c and major drawing/input components (Input.c, Report.c)

5. Bogus IPL to test DOL loading

6. DOL loading (Dolphin SDK demos) (All loaders are located in Machine.c)

7. GekkoCommon (especially Mmu)

8. GekkoInterpreter

9. Add some basic Flipper hardware to test simple demos. Get PONG work :=)

10. Write wireframe Flipper GX implementation to test simple GX demos

Still no sound. DSP will be the next stage.