fat
n02

by 0746

An Open Kaillera project

This is not an SRS or anything formal, just a documentation of my thoughts. For me, it helps to write down things so I don't loose sight of what is important and what is not.

Contents

1 Intro

Another prototype for the open kaillera client to explore some new concepts.

2 Purpose

This prototype intends to utilize/explore the the following concepts:

- exploring automatic code generation for cores
- 3+ players p2p
- non-linear predictor for delay masking
- variable delay?

3 Why its called fat

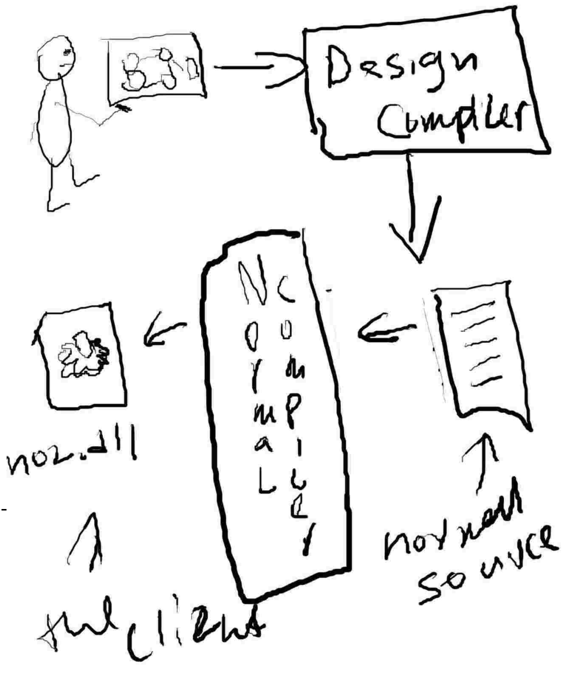
Something like non-linear predictor for delay masking algorithm is rather complex in terms of algorithemic time and space complexity compared to the constant time in my previous approach. With the philosophy of optimization in development that I've carried so far for this project, as lame and counterproductive as it was, I'd have never even considered something like this to exist in this project. My previous self would have called this "fat," hence the name fat n02.

4 before anything else: developing some sort of framework

As awful as my sense of professionalism is, I'd write to re-write the commpon dependent files completely to support the new ideas and stuff.

5 first: exploring automatic code generation for cores

The ammount of time I waste re-hacking stuff is extraordinary in ratio to other things I work on. Even when I make simple changes, I forget to add simple things and then if I'm lucky, someone reports a bug that I can trace to that broken thing. It's just not that, even when I want to test simple concepts, I end up re-coding a lot of stuff that are conceptually domain independent. Of course, the most obvious way to reduce them is to program in more systematic manner for abstraction and what not as opposed to trying to optimize things the lame way. But since we're not going to worry about that in this project, might as well try some things I learneded in school:



6 3+ players p2p

7 non-linear predictor for delay masking

8 variable delay?

9 Coding Standards

All source files shall have the following header elements at the begining of the source file:

- Name of source file
- Purpose of source fole
- Author name and copyright notice

10 Code Gen Input

- The input models for code gen models shall be divided into machines, each with distinguished names.
- Each machine may have several parts, each with distinguishing names.

Sample source file:

```
machine <MachineName>;
part <PartName> {
        [init] state <StateName> {
                entry {
                        ...
                }
                on (<EventName>(<EventParamDeclarations...>)) {
                        ...
                        transit <NewStateName>;
                }
                ...
                exit {
                        ...
                }
        }
        ...
}
```