SecHack365

コンピュータ技術をゼロから作り直す memepu

開発駆動コース　川合ゼミ　コートニー　エリオット

概要

ブラックボックス（ FPGA 等）を使わずコンピュータ技術を作り直すプロジェクト。

大まかな構造

低レイヤーから高レイヤーまでの予定する構造

ハードウェア（ kekpu: 開発中、 kekcad: 一旦開発中止、 kekroute: 一旦開発中止）

⇔ マイクロアーキテクチャ（ kekpu: 発中、 kekware: 開発完了）

⇔ISA （ kekpu: 開発中、 kekware: 開発完了、 keksim: 開発完了）

⇔ アセンブラ（ kekasm: 開発完了）

⇔ 自作言語（ keklang: 開発中、 kekpile: 開発中）

⇔OS （ kekos: 開発予定）

全体のセキュリティ問題について気づいたこと

- ステート数が多いため、大きなシステムを形式的検証するのが難しい
- パソコンは線形有界オートマトンでステート数が有限なので量子コンピュータによる形式的検証はありだろうか

セキュリティの要素・価値

層状セキュリティ

- 低レイヤーから高レイヤーまでの知識を集めてセキュリティ問題を予測する能力を身につける

単一障害点（ SPoF ）

- 技術（ OS や CPU アーキテクチャやスタンダードの実装）にはバリエーションがなければ壊れやすくなったりセキュリティ問題が生じる

The Ken Thompson Hack

- コンパイラウイルスを見つけるには、既存技術に依存しないコンピュータを作る必要がある

教材としての価値

- FPGA はブラックボックス、 memepu は透明性が高い
- 低レイヤーを深く理解したい人に向いている

kek**pu**

```
OP_LHU: begin
  `SET_MNEMONIC("lhu r%1%,%4$x")
  case (microop_count)
    0: begin
      reg_sel = REG_SEL_OPWORD0;
      out_plane = OUT_OPWORD_IMMEDIATE;
      in_plane = IN_REG;
    end
    1: `GO_FETCH()
  endcase
end
```

MVP 完成率： 90%

開発期間： 2019 年３月〜

Verilog で定義した CPU である。命令はマイクロコードによって実装している。

フィーチャ・特徴：

- ワードサイズが４バイトで、バイトアドレス不可
- ファームウェアを自動的に生成する
- 形式的検証で証明（一部のみ）
- 74 シリーズ IC と SRAM と EEPROM で実装
- 割り込み処理、マルチタスキング、タイマー

IPC	約 0.1 命令 / サイクル	命令は平均として 10 サイクルがかかる (スーパースカラーではない)
最大メガヘルツ	約 40 メガヘルツ	LVC ロジックと SRAM のスピードに限られている
メモリ容量	最大数百メガバイト	現在、 SRAM を使っている
32 ビットレジスタの個数	32 個	16 個なら足りない、 64 個なら機械語で 6 ビット必要となり割に合わない
必要な IC チップの個数	>90 個	全部、論理回路と SRAM とルックアップテーブルなので多い

kek**asm**

```
lhu r0,0 ;fib 0
lhu r1,1 ;fib 1
loop:
  add r2,r0,r1 ;compute next fib
  addu r0,r1,0 ;swap
  addu r1,r2,0 ;swap
  lhu r31,loop ;loop
```

MVP 完成率： 100%

開発期間： 6 月〜 8 月

kekpu が実行する機械語を作るアセンブラ。 RISC-V を参考にした ISA だが、 RISC-V より簡単で命令フォマットが少ない。

機能：

- 相対分岐、ラベル、データアセンブラ指示語
- kekpu から自動生成する JSON モデルによって新しく実装した命令がすぐ使える

INT	ソフトウェア割り込み	LHU rD, I	符号なし整数をレジスタに書き込む
SETI	割り込みを有効にする	LW rD, [rS + signext(I)]	メモリからレジスタに書き込む
CLRI	割り込みを無効にする	SW [rD + signext(I)], rS	レジスタからメモリに書き込む
SRL rD,rA,rB	右論理シフト	BEQ rA,rB,I	等しければ分岐する
LH rD, I	符号付整数をレジスタに書き込む	BNE rA,rB,I	等しくなければ分岐する
ADD rD,rA,rB	足し算	OR rD,rA,rB	ビット単位 OR
XOR rD,rA,rB	ビット単位 XOR	SLL rD,rA,rB	左論理シフト

kek**lang**

Rust と Go の中間のような言語。静的型付けで強いテンプレートのおかげでポインター演算などを標準ライブラリで実装している。インタプリタを C++ で書いて、今は kekpile というコンパイラを keklang で実装中。

MVP 完成率： 70%　開発期間： 6 月〜

kek**sim**

Verilator を通じて kekpu をシミュレートする。キーボードなどの入力は MMIO と割り込みで実装しており、 VRAM を直接表示に出している。

MVP 完成率： 100%　開発期間： 6 月〜

kek**route**

verilog で書いた CPU を自動的に回路図にするプログラム。リーフモジュールの中（ 74 シリーズ IC ）以外に配線の機能しか使っていないので、回路図にすることができる。これによって、ヒューマンエラーを防ぎ、早く開発し、大きなリファクタリングをすることが可能になる。

現在は、 kekpu の一部（ ALU ）を回路図にすることができてアイディアの可能性を証明した。

NICT 国立研究開発法人 情報通信研究機構

MVP 完成率： 30%　開発期間： 2019 年４月〜５月

kek**cad**

プリント基板を作るには回路図と基盤の設計が必要。回路図は kekcad によって自動的に生成できるが、基盤の設計は自分で作るのに手間がかかるため、自動的に設計しようとしている。現在、簡単なグリッドオートルータ（自動配線システム）になっている。ハードウエアを作る段階になったらトポロジカルオートルータを作りたい。

Verilog で定義した構造をうまく使えば、部品自体のレイアウト（配線だけではなく）ができるというアイディアもある。

MVP 完成率： 80%　開発期間： 2019 年 2 月〜