



RISC-V CPUチップ評価の実践と再現性検証

— FPGA・Google Open MPW、
TinyTapeoutによるJASA独自RISC-V
オープンプラットフォーム開発とGithub公開

2025年12月4日

JASA 技術本部ハードウェア委員会 RISC-V WGメンバ

1-1. 発表者紹介



About JASA

小檜山智久

全体概要

JASA RISC-V WG主査 ((株)日立産機システム)

黒川能毅

JASAチップ1の内容

WGメンバ 個人開発者

小林正隆

デモンストレーション

WGメンバ ((株)日立産業制御ソリューションズ)

河崎俊平

JASAチップ2のGithub公開

WGメンバ (SHコンサルティング)

1-2. JASAについて



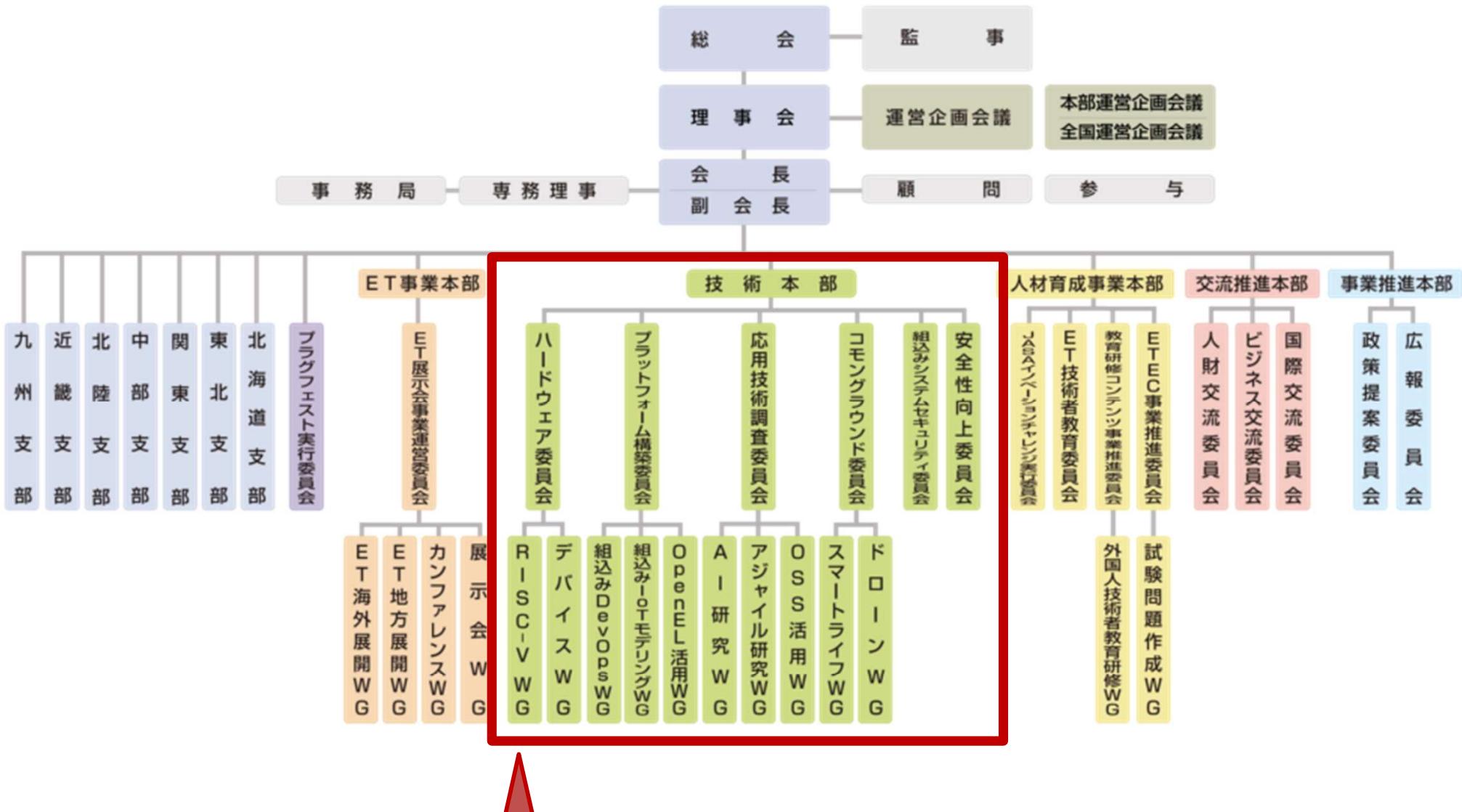
About JASA

名称	一般社団法人 組込みシステム技術協会 (Japan Embedded Systems Technology Association 略称「JASA」)
会長	竹内 嘉一
事務局	本部：東京都千代田区麹町一丁目入船 1-5-11 弘報ビル 支部：北海道、関東、中部、北陸、近畿、中国、四国、九州、沖縄
目的	<p>組込みシステム技術の普及と発展を目的とした組織として、技術を学ぶ場、技術を販売する場、技術を研究する場を提供する。また、技術者育成、技術開発、技術交流、技術情報の発信、技術規格の策定等を通じて、技術の発展に貢献する。</p> <p>（1）組込みシステム技術の普及と発展、地域振興等のための事業の運営及び貿易・技術の輸出入等の事業の運営</p> <p>（2）組込みシステム技術の研究開発、内外関係機関との情報交換並びに情報の提供</p> <p>（3）組込みシステム技術の研究開発、内外関係機関との情報交換並びに情報の提供</p> <p>（4）組込みシステム技術の研究開発、内外関係機関との情報交換並びに情報の提供</p> <p>（5）組込みシステム応用技術の研究開発、普及啓発</p> <p>（6）本会の会員に対する福利厚生に関する事業の推進</p> <p>（7）その他本会の目的を達成するために必要な事業</p>
会員数	正会員・支部会員：145社、賛助会員：28社、 学術会員：3団体、個人会員：9名（2023年4月1日現在）
設立	昭和61年8月7日（平成24年4月1日 一般社団法人へ移行）

1-3. RISC-V WGの位置づけ ①



Positioning of RISC-V WG



1-4. JASA RISC-V WGの活動について



About the activities of JASA RISC-V WG

《WGの活動方針》

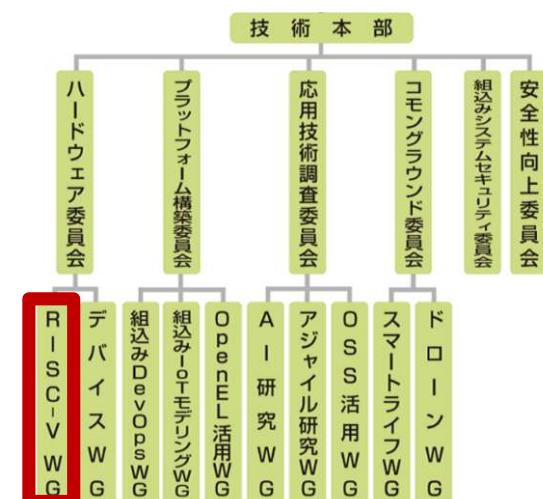
- ・オープンな仕様で会員が自由に活用できるRISC-Vプラットフォームを会員の協力で整備し、組込み分野でのRISC-V普及に努める
- ・関連団体とのコラボによりプラットフォームの応用範囲を広げる

《活動内容の項目》

- ◆ 月例WGの開催
- ◆ RISC-V著名人を講師にお迎えし、Webセミナーを開催
- ◆ 組込みに使えるRISC-Vプラットフォームの整備
- ◆ RISC-V関連団体との協創

《WGメンバ》

- ・委員：15社, 3校、26名



1-4. JASA RISC-V WGの活動について



About the activities of JASA RISC-V WG

《WGの活動方針》

- ・オープンな仕様で会員が自由に活用できるRISC-Vプラットフォームを会員の協力で整備し、組込み分野でのRISC-V普及に努める
- ・関連団体とのコラボによりプラットフォームの応用範囲を広げる

《活動内容の項目》

- ◆ 月例WGの開催
- ◆ RISC-V著名人を講師にお迎えし、Webセミナーを開催
- ◆ 組込みに使えるRISC-Vプラットフォームの整備
- ◆ RISC-V関連団体との協創

1-5. 開発ロードマップと実績

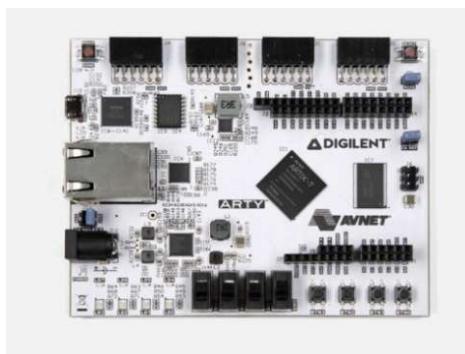


Development schedule for last 6 years

《過去6年間の活動》

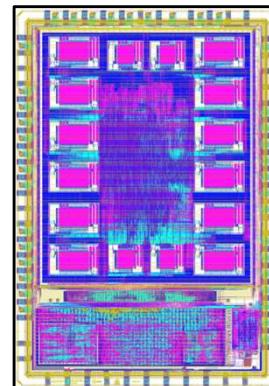
2020-22年度	2023年度～
<ul style="list-style-type: none">・32ビット版(Arduino)/64ビット版(LINUX)RISC-VコアFPGA実装・产学連携・成果のWeb公開	<ul style="list-style-type: none">・JASA版RISC-V SoC開発<ul style="list-style-type: none">- ターゲット:セキュアなIoTエンジン- オープンシリコン開発環境の活用・ボード作成, RTOS実装・IoTクラウド接続

FPGAベース



市販FPGA評価
ボード Arty A7
35T:32/Arduino
100T:64/LINUX

カスタムLSI(SoC)ベース



eFabless
chipignite
(…で計画)

1-6. RISC-Vプラットフォーム開発の狙い



Aim of developing the RISC-V platform

《初めて取り組む人の課題》

- ・オープンなのでどこかに情報は公開されている
- ・それぞれの場所に、それぞれの言語で点在する情報を丹念に集め、断片をつなぎ合わせ、試行錯誤しながら理解を深めて作り上げるには時間と執念とある程度の知識が必要
⇒ 初心者にはつらい！

《WGが提供する価値》

- ◆ WGメンバーで実際に検証して実績がある手順を確立
- ◆ 日本語/英語のバイリンガルで参照できるWebページの公開
- ◆ 開発環境の準備から完成まで一気通貫で説明
- ◆ 困ったらWGのオブザーバになってわからないことを聞ける
- ◆ 開発環境等の更新にできる限り対応
- ◆ 追試したら自動的に自分用のプラットフォームになる

1-7. 開発ロードマップ



Development roadmap

《長期計画》

JASAチップ[°]1

RoT外付
アナログ外付
無線通信外付

デュアルRISC-V
IoT管理チップ

2023
eFabless 130nm

JASAチップ[°]2

RoT内蔵
アナログ内蔵
無線通信外付

デュアルRISC-V
IoT管理チップ
RoT含

2026
AiSol
フラッシュプロセス

JASAチップ[°]3

RoT内蔵
アナログ内蔵
無線通信内蔵

IoT管理チップ
RoT含
無線通信含

2029
AiSol
フラッシュプロセス

RoT: Root of Trust

1-8. 開発のスコープ[°]



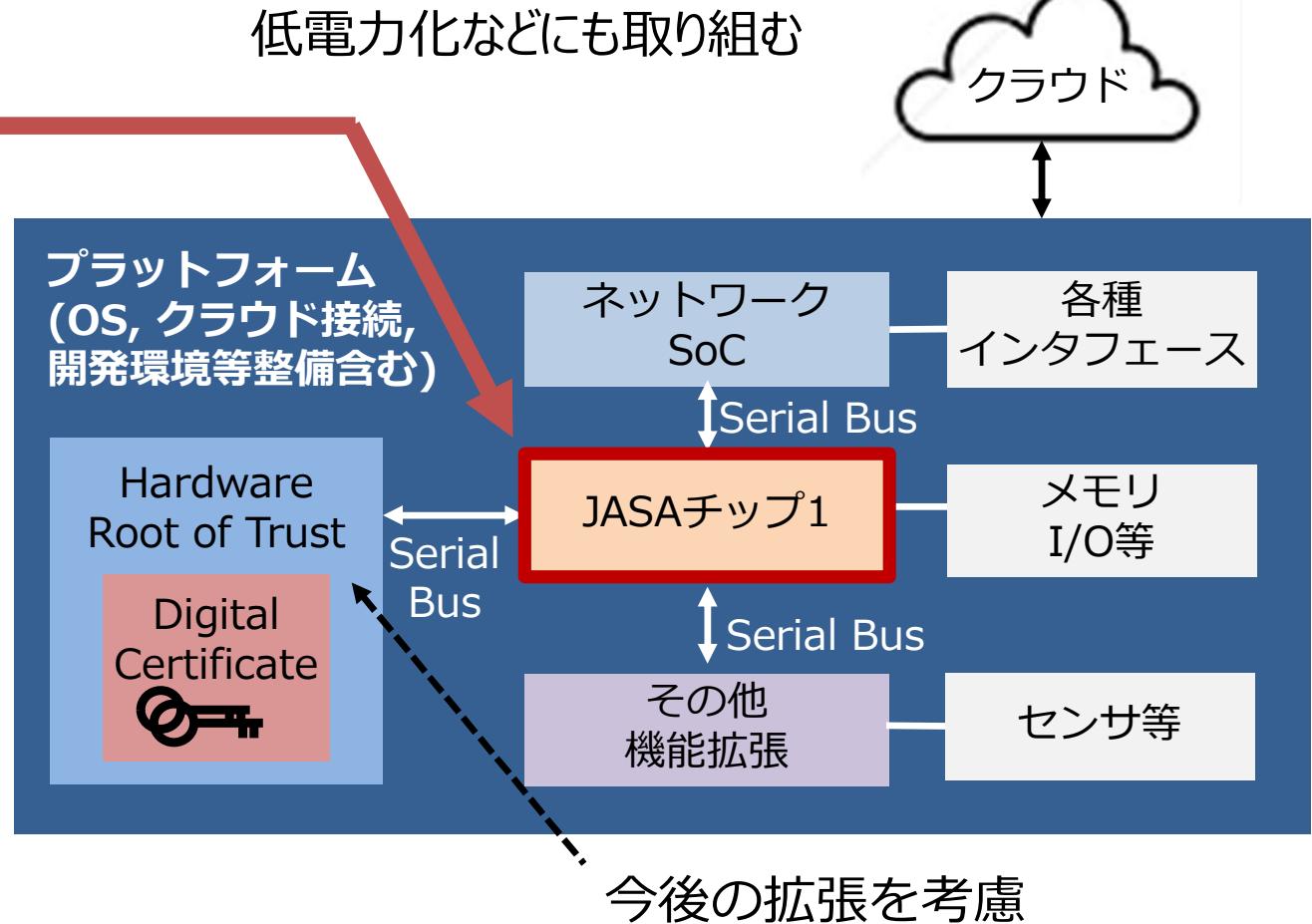
Development Scope

《IoTシステム化までがターゲット》

JASAチップ[°]1
RoT外付
アナログ外付
無線通信外付

デュアルRISC-V
IoT管理チップ

2023
eFabless 130nm



RoT: Root of Trust

1-9. 開発ロードマップ



Development roadmap

《長期計画》

JASAチップ¹

RoT外付
アナログ外付
無線通信外付

デュアルRISC-V
IoT管理チップ

2023

eFabless 130nm

JASAチップ²

RoT内蔵
アナログ内蔵
無線通信外付

デュアルRISC-V
IoT管理チップ
RoT含

2026

AiSol
フラッシュプロセス

JASAチップ³

RoT内蔵
アナログ内蔵
無線通信内蔵

IoT管理チップ
RoT含
無線通信含

2029

AiSol
フラッシュプロセス

コレ

- ・eFabless活動停止によりTinyTapeout用に軽量プロセッサを開発し、新JASAチップ¹とする
- ・従来検討していたJASAチップ¹(Marmot)に今後RoTを実装してJASAチップ²とする

RoT: Root of Trust



2. Tiny Tapeoutを活用した 独自RISC-V CPUチップの試作

2-1.ベースとなるCPU：教科書通り



Base CPU: By the book

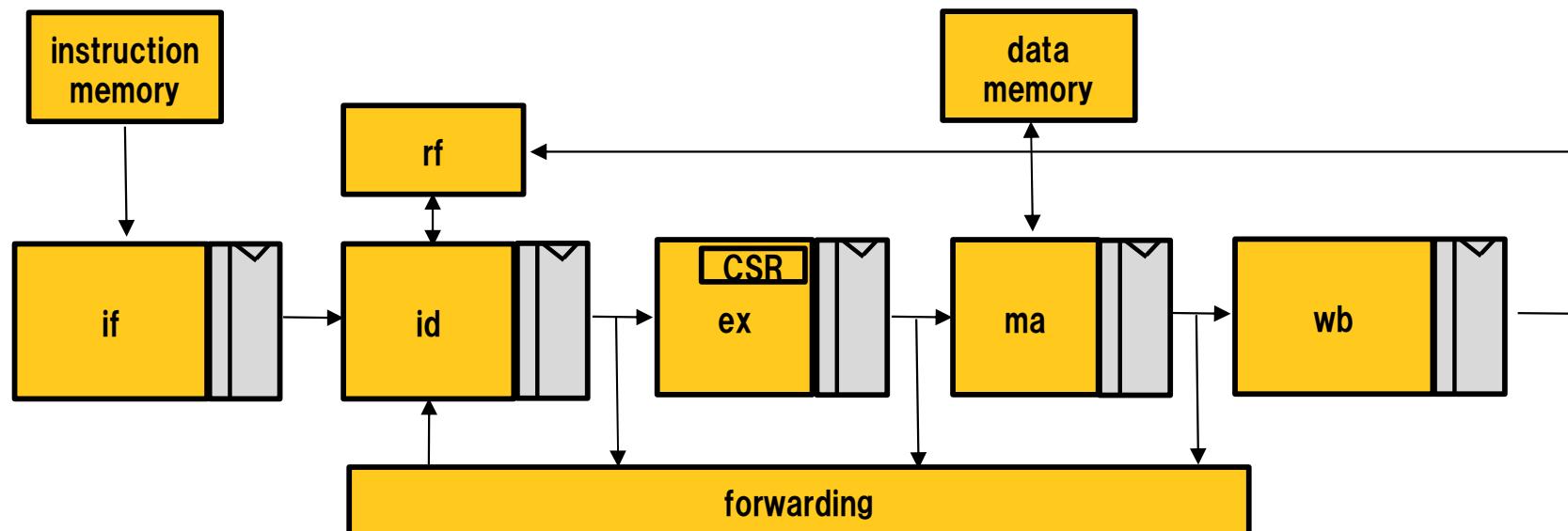
RISC-V CPU 個人開発の経緯

- ・2020年春コロナ禍初GW、1週間の休みでも外出できない
- ・時間つぶしのためCPUでも作ってみようか？

単純な5段パイプラインRISC-VをVerilogで書いてみよう…

⇒ 沼の始まり

- ・教科書通りのハーバード・アーキティクチャのCPUをコーディング
- ⇒ その後時間のある時に細々とFPGAデバッグをすすめる



2-2. Tiny Tapeoutについて



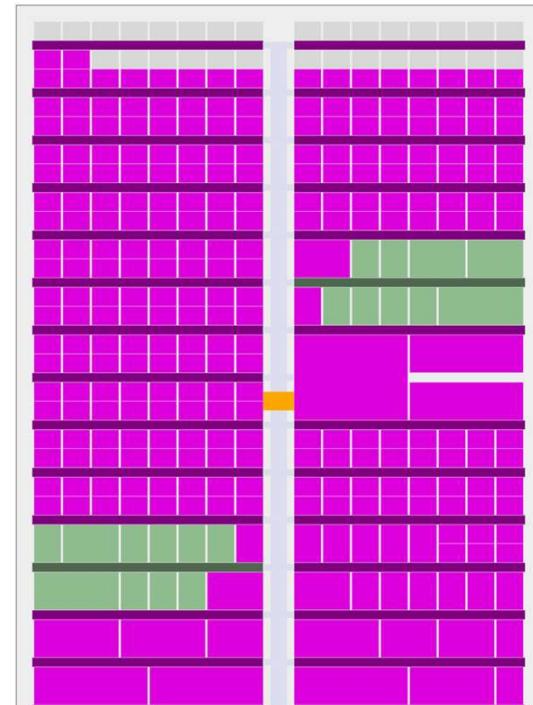
About Tiny Tapeout

Tiny Tapeout : 回路デザインを実際のチップ上に製造することを
簡単かつ安価に行うことができる教育プロジェクト

チップ上のデザイン可能領域を $167 \times 108 \mu\text{m}$ の微小なタイル領域に分割してタイル毎に契約することで、多数のデザインを 1 チップに相乗りさせ、またDAツールとしてオープンソースのツールチェイン(Open Road)を利用してことで、安価な試作チップ作成フローを実現。

Fabとしては Skywater社 sky130など
オープンソースとなっているPDKを使用

<https://tinytapeout.com/>



2-3. Tiny Tapeout試行



Tiny Tapeout Trial

2025年春、JASA RISC-V WGにて：

前述のTiny Tapeoutのサービスを知る

果たしてTiny Tapeoutで自作CPUは作れるのか？

手元の5stage RISC-VでTiny Tapeoutを試行

面積削減のため命令,データメモリを64ワードまで減らして試す

結果：最大のタイルサイズ8x2に対して4.6倍のサイズとなり

遠く及ばず（1タイルサイズ = $167 \times 108 \mu\text{m}$ ）

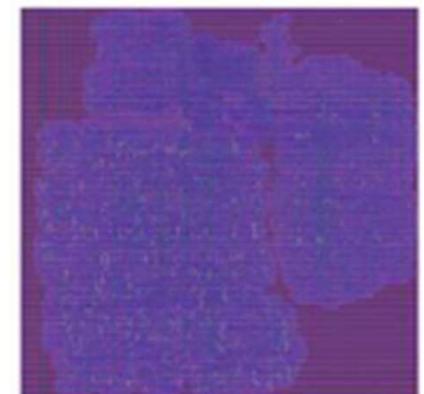
原因： ①メモリ5184ビット全てがFlipFlop(FF)となる

②また約9000超ビットがパイプラインFFなどで存在

⇒ 少なくとも今の論理で入りきるものは作れず、

非パイプライン型の小さなCPUを設計することを決心

8x12サイズに無理やり入れると右のようなレイアウトになる ⇒



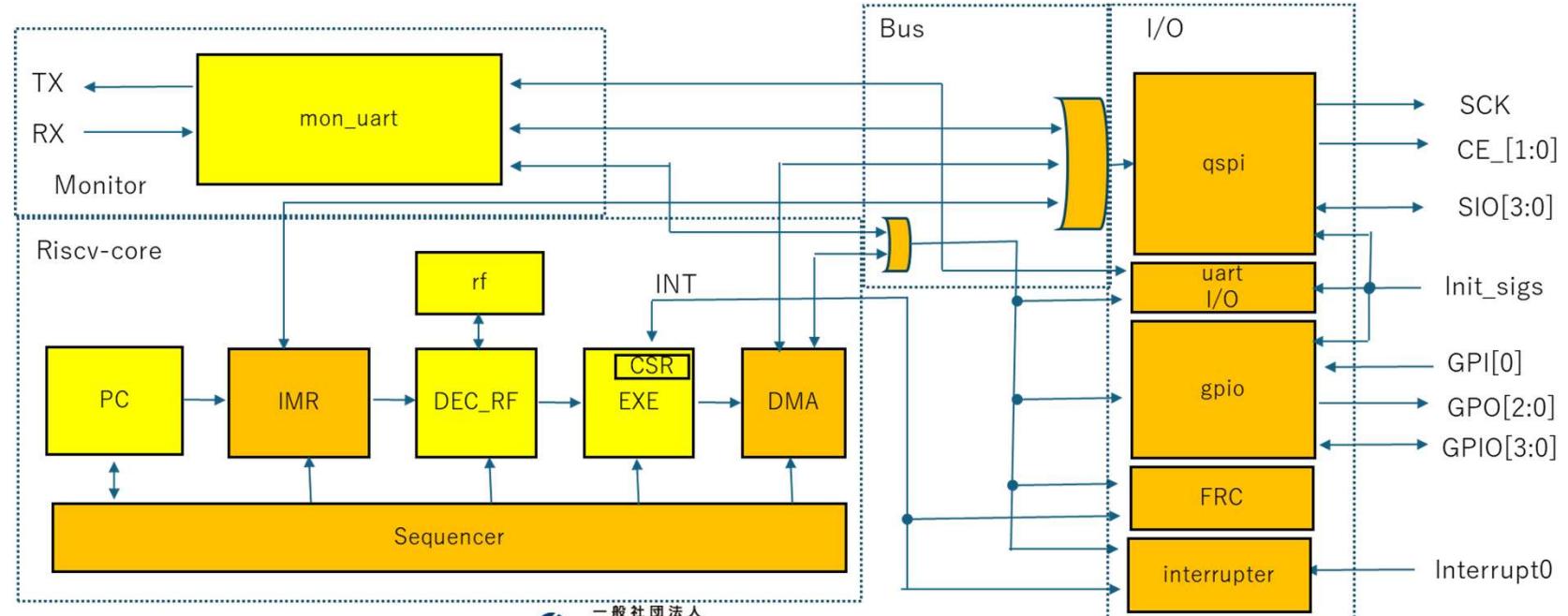
2-4. 省回路のCPU : State Machine方式



Small Area CPU : State Machine Type

以下の方針でCPUを組みなおす

- ・ステートマシンで実行ステージを決定しそのステージを処理
⇒ ステージ間のFFを削除
- ・命令,データメモリを外部のQSPIメモリに代用させ、メモリはRFのみ
⇒ メモリFFを大幅削減
- ・できる限り 5stage版の論理を活用する … 実装期間の短縮
- ・テストベンチも5stage版を極力利用する … テスト期間の短縮



2-5. 論理セル数比較



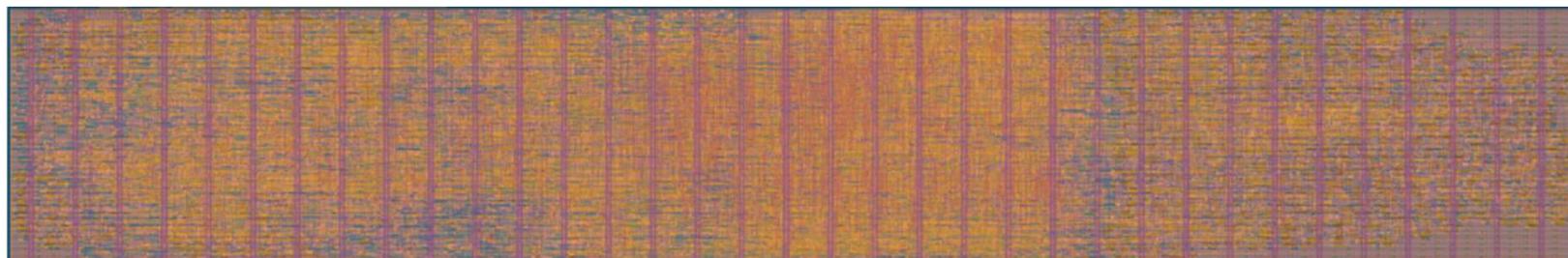
Comparison of Number of Logic Cells

設計と実装：個人開発のため、2025/7月の土日祭日と早朝を使い
約2週間でざっくりFPGAテストまで進める

⇒ Tiny Tapeoutのフローに乗せてみる
8x2のサイズ化に成功：5Stage版の1/6の面積サイズ

表1 論理セル数比較

#	セル種別	5stage	State machine	倍率
1	メモリ部分FF	5184	1088	1 / 4.76
2	それ以外のFF	9334	1616	1 / 5.78
3	ロジックゲート	63541	9543	1 / 6.65



2-6. 開発CPUスペック表



Specification Table of Developed CPU

項目	内容
命令セット	RV32I (RISC-V命令セットの最小構成)
プロセッサ構成	State Machineタイプのシングルプロセッサ
動作モード	Mモードのみ
搭載メモリ	なし 外付けメモリを使用
サイズ	1336 μm x 216 μm
ゲート規模	約40 KGates (2nand換算)
目標動作周波数	25~50MHz (ただしI/Oの影響で低下する可能性あり)
信号ピン	8 input 8 output 8 bi-direction
外付けメモリ	QSPI PSRAM/Flash ROM
タイマー	64bitフリーラン
I/O	4bit GPIO, SPI, UART, 割込み1ピン
モニター機能	UARTにハード組み込み

2-7. モニター機能



Monitoring Function of CPU

初代のCPU時代に開発

- ・フルVerilog簡易モニター機能をUARTに搭載

・実装機能（最低限デバッグに必要な機能）

- メモリのリード・ライト
- CSR/RF/IOLレジスタのリードライト
- CPU実行、停止
- 現在アドレスの表示
- ブレイクアドレスの設定・解除
- CPU実行時はCPUのUARTとして機能

・簡易機能であるが、FPGA使用の
実機テストで威力を發揮

The screenshot shows a terminal window titled "COM5 - Tera Term VT". The window displays a series of text-based commands and responses. The commands include "g00001000", "Hello FreeRTOS!change value3", "change value", "uxTimerIncrementsForOneTick 326797", "change value2", "change value6", "1 : 0 : 3700 : change value4", "0: Tx: send 1", "0: Rx: received 1", "0: Tx: send 2", "0: Rx: received 2", "0: Tx: send 3", and "0: Rx: received 3". Below these, there are several memory dump entries starting with "p c000 f800 c000 f820" followed by hex values like "1b820f7b", "00000000", "0aa819ef", "00000000", "00000005", "deadbeef", "deadbeef", and "deadbeef".

2-8. Tiny Tapeout向け検証



CPU Testing for Tiny Tapeout

以下の検証環境を整備

- ・FPGA上の開発環境一式を5 stage版流用
- ・自作のRISC-V RV32Iのアセンブラー
- ・risc-v toolchain Cプログラムでベアメタル動作する環境を構築
- ・I/Oやタイマー、割り込みについてはCでテスト作成
⇒ FPGA向けの動作を楽しむ個人開発環境としてはOK

これだけでは修正不可能なASICの検証としては不足

しかし、プロ同様の多量の検証ベンチセットの準備は到底不可能

→ 目標をFPGA上でOSSのFreeRTOSのデモを動かす事に設定
理由：動作要素として、ecallやタイマー割込みなどが入っており、
挙動が複雑

2-9. FreeRTOSとの格闘



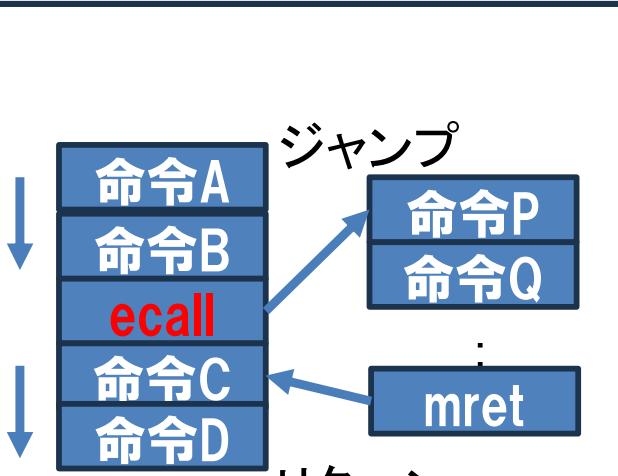
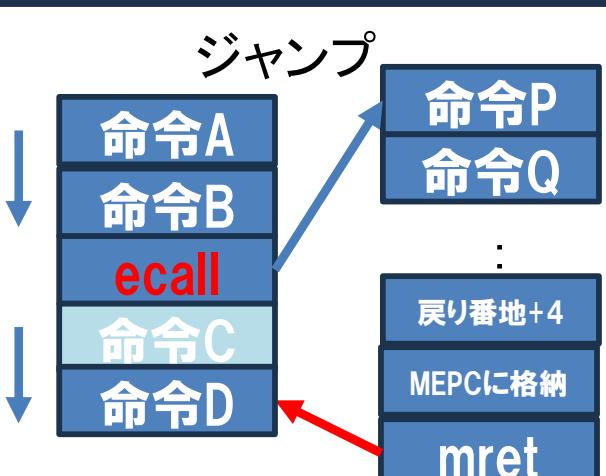
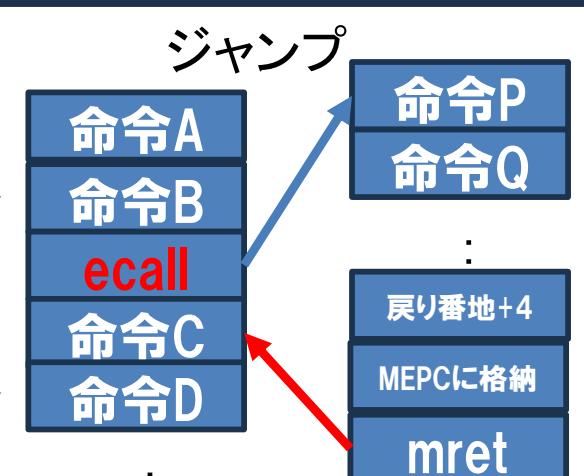
Struggle of FreeRTOS on Developed CPU

発見した不具合：条件により数秒～数時間で停止

原因：ecallの仕様の理解の間違い(詳細は下図)

仕様をきちんと理解していれば避けられたのですが、
ゆるい個人開発では抜けが出てしまったのでした…

⇒ この不具合修正によりFreeRTOSデモの1日連続動作を確認

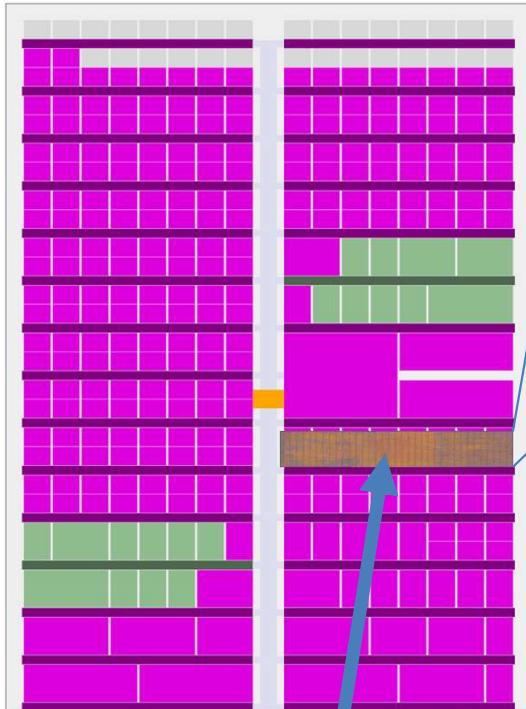
想像していた ecallルーチンの動作	実際のecallルーチンの動作 (命令C実行されない)	不具合修正後の ecallルーチンの動作
 <p>命令A 命令B ecall 命令C 命令D</p> <p>ジャンプ</p> <p>mret</p> <p>リターン</p>	 <p>命令A 命令B ecall 命令C 命令D</p> <p>ジャンプ</p> <p>戻り番地+4</p> <p>MEPCに格納</p> <p>mret</p> <p>リターン</p>	 <p>命令A 命令B ecall 命令C 命令D</p> <p>ジャンプ</p> <p>戻り番地+4</p> <p>MEPCに格納</p> <p>mret</p> <p>リターン</p>

2-10. JASAチップ1の全体構成



Summary of R&D status in FT2024～

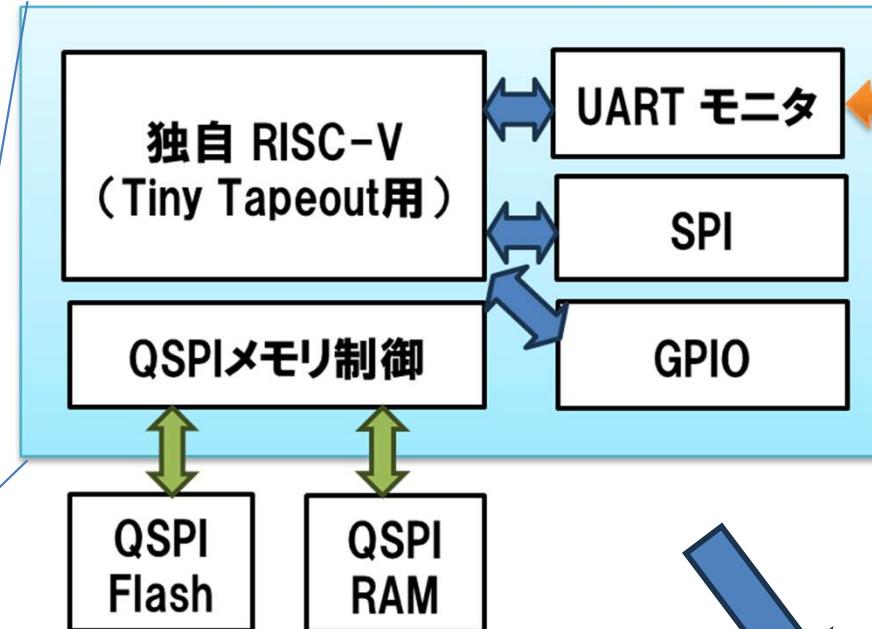
Tiny Tapeoutチップイメージ



JASAチップ1のレイアウトイメージ



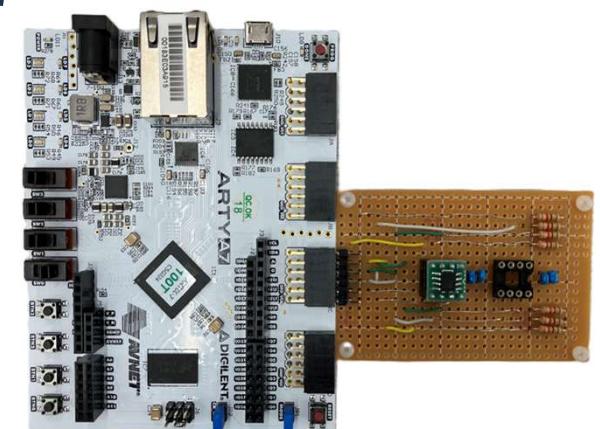
JASAチップ1の構成ブロック図



PCから制御

- プログラムロード
- プログラム実行
- プログラム停止
- I/Oリード・ライト
- メモリリード・ライト
- レジスタダンプ

FPGAの検証環境構築



同一Verilog
から生成

2-11. Tiny Tapeout向けCPU開発まとめ



Conclusion of Small Area CPU Development

結論

- ・Tiny Tapeout使用可能なRISC-V(RV32I)互換CPUを個人で開発
- ・State Machineタイプを選択し、約1/6に論理サイズを縮小
- ・他開発の部品とテストプログラムを再利用し開発工数を削減
- ・FreeRTOSをポーティングし論理テストの網羅性を少しでもカバー

今後の展開

- ・2026/3のTiny Tapeoutを目指して実機検証を継続
- ・SDカードサポート（超低速でもストレージをサポートできないか？）
- ・そのほか未対応のCSRのサポート



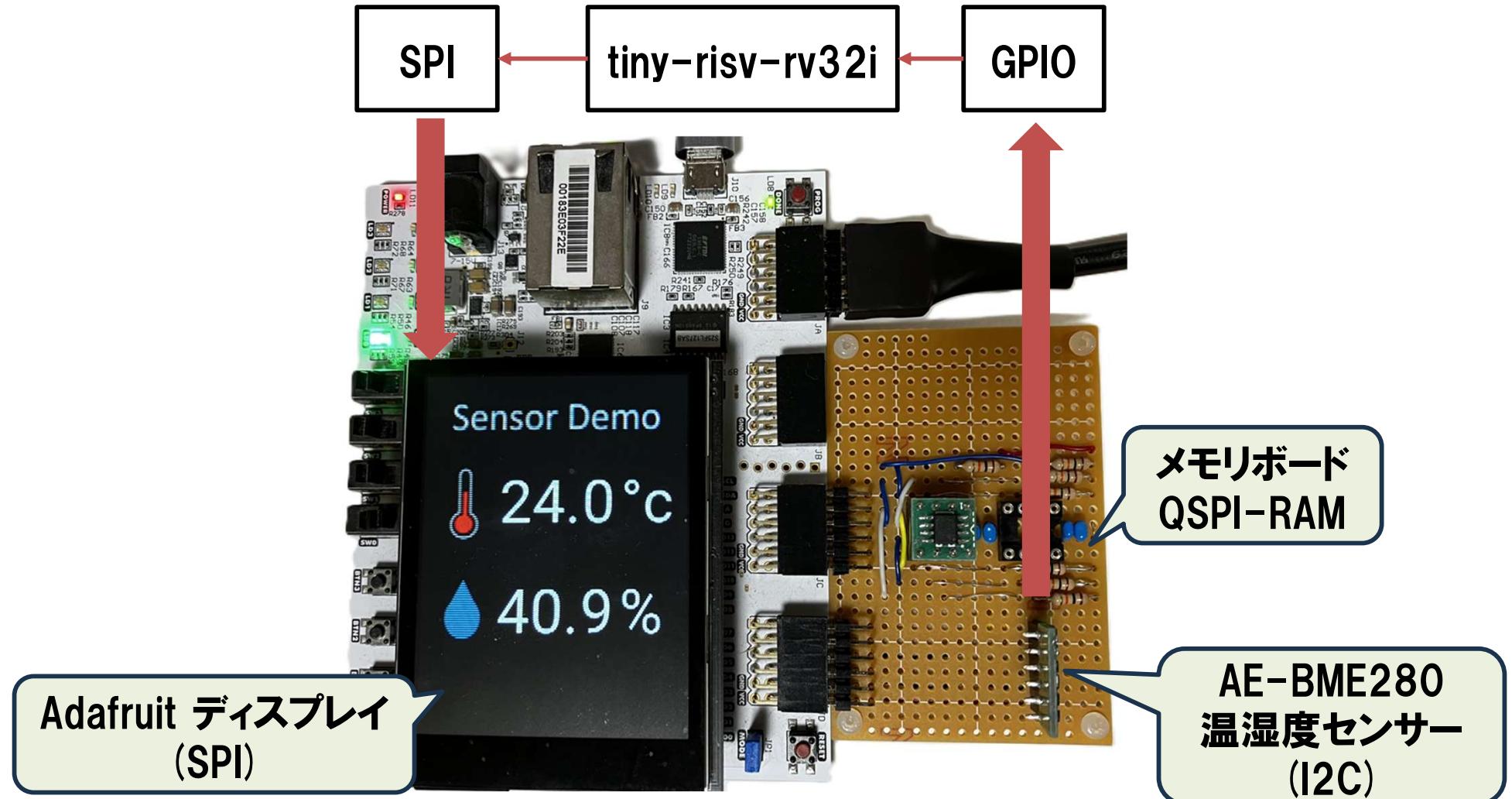
3. デモンストレーション

3. デモ概要



Summary of Demonstration

JASAチップ1の tiny-riscv-rv32i で動作するデモを作成しました。





4.

Marmotのソース公開と 今後の展開に向けた課題

- ・eFabless活動停止によりTinyTapeout用に軽量プロセッサを開発し、新JASAチップ1とする
- ・従来検討していたJASAチップ1(Marmot)に今後RoTを実装してJASAチップ2とする

4-1. Marmotのソース公開



Marmot Source Release

以下のサイトで公開

https://github.com/OpenProcessors/marmot_asic

SiFive FreedomをベースにCaravel 対応

- testcase/hello correctly ran on SPI Flash!
- .
- Trying to run testcase/hello
- Add sim/
- ECP5-LFE5UM5G-85F worked with inferred SRAM (implemented with FFs)
- Infer SRAM
- .
- ECP5-LFE5UM5G-85F worked!
- top.lpf -> ECP5-LFE5UM-45F.lpf
- Change scripts/vlsi_rom_gen for FPGAs implementation
- E300 on Versa-ECP5 run bootrom/led correctly git status
- E300 on Versa-ECP5 run bootrom/led correctly git status
- Add bootrom/led for E300
- Add E300 Versa ECP5 build env
- Update README.md
- Update README.md
- Merge pull request #146 from Tim453/e310_with_e31-core
- Added BTB and a 16kB 2-way I-Cache
- Merge pull request #136 from mct/
- Reduce time to install packages
- Merge pull request #118 from mct/
- Bump fpga-shells, supports vc707 with and without FMC-PCIe module
- Revert url of submodule for pull request
- Added support building for vc707 with/without PCIe HMC module

Marmot としてのカスタマイズ

SiFive Freedom

- shc <kesami.hagiwara@swihwc.com>
- Erik Danie <43764516+erikdanie@users.noreply.github.com>
- Erik Danie <43764516+erikdanie@users.noreply.github.com>
- Erik Danie <43764516+erikdanie@users.noreply.github.com>
- Tim <y0083012@tu-bs.de>
- gerdes@sifive.com>
- moto@gmail.com>
- anie@users.noreply.github.com>
- Akira Tsukamoto <akira.tsukamoto@gmail.com>
- Akira Tsukamoto <akira.tsukamoto@gmail.com>
- Akira Tsukamoto <akira.tsukamoto@gmail.com>

4-1. Marmotの課題



Issues of Marmot

Marmot には Freedom と同じ課題が存在

1. submodule のURLが古い
2. 古いビルド環境が必要

他にも課題があるかもしれない

4-3. 課題① submoduleのURLが古い



Issue of Marmot #1: Expired URL of Submodule

課題

- submodule のURLが古い

path	現状	修正
roms/vgabios	git://git.qemu-project.org/vgabios.git/	https://github.com/qemu/vgabios.git
roms/seabios	git://git.qemu-project.org/seabios.git/	https://github.com/qemu/seabios.git
roms/SLOF	git://git.qemu-project.org/SLOF.git	https://github.com/qemu/SLOF.git
roms/ipxe	git://git.qemu-project.org/ipxe.git	https://github.com/qemu/ipxe.git
roms/openbios	git://git.qemu-project.org/openbios.git	https://github.com/qemu/openbios.git
roms/openhackware	git://git.qemu-project.org/openhackware.git	https://github.com/qemu/openhackware.git
roms/qemu-palcode	git://github.com/rth7680/qemu-palcode.git	https://github.com/qemu/qemu-palcode.git
roms/sgabios	git://git.qemu-project.org/sgabios.git	https://github.com/qemu/sgabios.git
pixman	git://anongit.freedesktop.org/pixman	https://gitlab.freedesktop.org/pixman/pixman.git
dtc	git://git.qemu-project.org/dtc.git	https://github.com/qemu/dtc.git
roms/u-boot	git://git.qemu-project.org/u-boot.git	https://github.com/qemu/u-boot.git

対策案

- submodule を取得するためのスクリプトファイルを準備

4-4. 課題② 古いビルド環境が必要



Issue of Marmot #2: Keeping Old Development Environment

課題

- 古いビルド環境が必要 (Ubuntu 18.04)

必要なツール

名称	version	説明
sbt	1.2.1	Chisel のビルドに必要 バージョンの特定: scala 2.12.4に対応したバージョンを検索
Java (openjdk)	11	バージョンの特定: 以下のサイトに記載あり https://docs.scala-lang.org/overviews/jdk-compatibility/overview.html Ubuntu 18.04 では、openjdk-11をapt-get で取得可能
verilator	3.922	Verilogのシミュレーション用 バージョンの特定: README.md に記載あり
RISC-V toolchain	10.2.0	バージョンの特定: SiFive社のツールチェーンの最新版 https://github.com/sifive/freedom-tools/releases

対策案

- ビルド環境を構築するためのスクリプトを準備

4-5. 活動成果の公開



Publication of Development Results

◆ 活動成果へのアクセス方法

- ・RISC-V WGのホームページ最下部のリンクをクリック

RISC-V
WGページ

IRISC-V WG

1. 活動報告(2023年度の事業方針)
2. 事務連絡

RISC-V WG - 最新情報

日本語リンク
English link

- ・日本語リンク
- ・英語リンク

プロジェクトリスト

1. RISC-V Linux in LiteX/Rocket on FPGA Arty A7-100T

2. Caravel SoC 事前評価

- ・FPGAリンク
- ・SoCリンク

Caravel SoC 事前評価

1. [A7A10] SoC ハードウェア開発環境 - Caravel SoC

2. [A7A10] SoC ハードウェア開発環境 - Caravel SoC

3. eFabless MPW で実現した - Caravel SoC

4. Sifive FreedomCA 基板 / Arty-A7-100Tコントローラ

- ・詳細項目リンク

[A7A10] SoC ハードウェア開発環境 - Caravel SoC

1. ソフトウェアのセットアップ

2. ハードウェアのセットアップ

3. ハードウェアの確認

スマホをお持ちの方はこちら ⇒





RISC-V CPUチップ評価の実践と再現性検証

2025/12/4 発行

発行者 一般社団法人 組込みシステム技術協会
東京都 中央区 入船 1-5-11 弘報ビル5階
TEL: 03(6372)0211 FAX: 03(6372)0212
URL: <https://www.jasa.or.jp/>

本書の著作権は一般社団法人組込みシステム技術協会(以下、JASA) が有します。
JASAの許可無く、本書の複製、再配布、譲渡、展示はできません。
また本書の改変、翻案、翻訳の権利はJASAが占有します。
その他、JASAが定めた著作権規程に準じます。