Linux が稼働する RISC-V ソフトプロセッサの FiC FPGA ボードへの移植

並木美太郎 [1]　三好健文 [2]　天野英晴 [3]

概要：近年、RISC-V と呼ばれるオープンソースのプロセッサが FPGA(Field Programmable Gate Array)と呼ばれるユーザが書き換え可能なハードウェアで利用できる。このようなハードウェアプラットフォームをシステムソフトウェア研究のプラットフォームとして利用したい。本報告では東工大吉瀬研究室が開発した Linux が稼働する RISC-V ソフトプロセッサを標準バスの AXI バスを用いるように修正し、共著者の天野研究室で開発した FiC(Flow-in-Cloud)ボードに移植した。

キーワード：RISC-V Linux FPGA FiC

Porting RISC-V Soft Processor Running Linux to FiC FPGA Board

Mitaro Namiki[†1]　Takefumi Miyoshi[2]　Hideharu Amano[†3]

Keywords: RISC-V Linux FPGA FiC

１.はじめに

近年、RISC-V[1]と呼ばれるオープンソースの CPU が FPGA(Field Programmable Gate Array)と呼ばれるユーザが書き換え可能なハードウェアにおけるソフトコアとして利用できる。本報告では、このような FPGA ボードの一つである FiC ボードをシステムソフトウェア研究のプラットフォームとして利用するため、東工大吉瀬研究室が開発した Linux が稼働す RISC-V ソフトプロセッサ[2-6]を、標準バスの AXI バスを用いるように修正して移植した。本報告では、その詳細、評価について述べる。

２．本研究開発の目的

OS などのシステムソフトウェアの研究では、新しいハードウェアに対する仮想化、資源管理の方式を考察したい。また既存の CPU の拡張を行って方式の有効性と性能向上を考察したい要求がある。チップを作るのは手間と予算がかかる。近年 FPGA(Field Programmable Gate Array)と呼ばれるロジックをある種のプログラム言語で記述し、ハードウェアを実現するチップが広く利用できるようになった。FPGA はチップ開発のテストデバッグだけでなく、計算アクセラレータでも使われている。

1. 東京農工大学
TokyoUniversity of Agriculture and Technology
2. わさらぼ合同会社
WasaLabo, LLC.
3. 慶応大学
Keio University

このような、FPGA を用いて、OS 研究の実行基盤にしたいが、修正したい CPU の命令セット、機能、記述には著作権があり、知財の問題から、オープンソースとして使えない、

このような問題から近年はロイヤリティフリーの命令セットアーキテクチャとして、RISC-V が着目されている[1]。また、RISC-V アーキテクチャを実現した FPGA 向けのオープンソースのソフトプロセッサがいくつか存在する。組込みソフトウェア向けの RISC-V はオープンソースの行数は少なく理解は容易だが、Linux が稼働する RISC-V は、数万行を超える。

筆者は OS の研究者なので、比較的容易に CPU を修正して検証できる環境を入手したい。

従って、

- 実行性能追及でなく、理解が容易なできるだけ小規模のオープンソースであること
- Linux を実行できるようにするためページングをサポートし、多重仮想アドレス空間を実現できること
- 主記憶や I/O デバイスの接続が容易であること
  できるだけ、標準部品で構成できること
- FPGA の使用リソースは応用処理も実現できるくらいの規模であること
- RISC-V の記述言語は、余計な学習、環境構築を避けるため、Verilog などの HDL であること。

筆者らの一人は、MEC（Multi-access Edge Computing）用の FPGA クラスタの研究開発を行っており、FiC（Flow-in-Cloud）はこのための構成要素となるボードである。FiC は、制御用に Raspberry-Pi3 のドーターボードを搭載しているが、FPGA 上に搭載する様々なハードウェアと密結合するためには、同一 FPGA 上にソフトプロセッサを搭載するのが有利である。

Digilient 社のボードなど、RISC-V を搭載することのできる FPGA ボードは数多いが、FPGA クラスタの構成要素となるボードや FPGA クラスタそれ自体に RISC-V を搭載した例は報告されていない。そこで、本報告では、RISC-V を FiC に移植し、Linux を実行することとした。

そこで、上記の条件を満たすオープンソースの RISC-V のコードを検討した。Linux を実行できるオープンの RISC-V のソースコードとして、CVA6、日本国内で開発された RISC-V として、東工大吉瀬研究室の RV-PC を検討した。どちらも VerilogHDL で記述されている。
RV-PC は 10700 行程度、CVA6 は 19000 行程度である。

２．RV-PC について

RV-PC は、東工大吉瀬研究室で研究された Linux が動くオープンソースのコードである。　東工大吉瀬研究室でアーキテクチャ教育を目的に研究開発された RV-PC は Verilog HDL で記述された 32 ビットの RISC-V である、詳細は文献[2-6]を参照されたい。５ステージパイプライン版 RV-PC と遂次実行 RVSoC と二つの版が存在する。どちらも Linux が稼働する。RV-PC は

Digilient 社 Nexsys4 DDR

と

Digilient 社　Arty A7-35T で稼働する。いずれも Linux カーネルが動作する。
遂次版は Verilog で 6300 行程度、パイプライン版は 10700 行程度である。

Nexsys4DDR は、VGA, イーサネット、SD カードコネクタを有し、USB マウスと pmod PS/2 を用いて PS2 キーボードを接続し、Xwindow を実行できる。Arty については、SD カードと GPIO イーサネット PHY カードを接続し、シリルコンソールで Linux を実行できる。
この RISC-V を FiC ボードに移植する。

３．FiC(Flow-in-Cloud)

FiC は、NEDO プロジェクトの研究成果として開発された、FiC(Flow-in-Cloud)は、中規模な FPGA ボードをシリアルリンクにより密結合して FPGA クラスタを構成し、仮想的に大きな FPGA として扱えるシステムの構築を狙っている[7,8]。FPGA は Mid-range のシリーズ中で高いコスト性能比を持つ Xilinx 社 XCKU115 を用いており、この上にマルチボード接続用のスイッチ、シリアルインターコネクト IP、DRAM コントローラを搭載する。ユーザが HLS または HDL で設計したアクセラレータは、部分再構成を使うことで、スイッチ部を変更せずに実装することができる。DRAM としては、DDR-4 SDRAM 4GB を 2 組、8.5Gbps のシリアルリンクを 32 組持つ。制御用の Raspberry Pi3 はドーターボードの形で搭載され、GPIO を用いて FPGA 内のロジックとデータ交換を行う。
なお、本ボードに基づく M-KUBOS は、Paltek 社より販売されている。

図 1 .FiC ボード


図２．Fic ボードを接続した FPGA クラスタ

４．目標

本研究開発では、RV-PC の移植性を向上し、FiC ボードをはじめとする各種 FPGA プラットフォームでのソフトプロセッサ RISC-V の稼動を可能とし、OS などのシステムソフトウェア研究の基盤とする。

５．RV-PC の移植

RV-PC の RISC-V を FiC ボードに移植する。
移植の際の主な作業は、
1)主記憶の DRAM
2)I/O
が問題となる。DRAM については、RV-PC は Xilinx 社の MIG(Memory InterfaceGeneraror)を用いる構成となっており、2)の I/O については、
RV-PC は、
a) UART
b) SPI-μ SD
c) USB mouse
d) Pmod　PS2 KB
e) GPIO Ethernet PHY
を有する。これらは Nexsys4 DDR ボードで標準的に利用できる。

オリジナルの RV-PC において、I/O は VirtI/O と呼ばれる FIFO を用いた仮想通信路で CPU と通信する。この構成を継続して利用する。
特に、IOProcessor と Main Processor の dual processor 構成となっており、IOP は VirtI/O をアクセス可能である。
移植で手間がかかるものは、主記憶の DRAM と I/O である。オリジナル RV-PC は、MIG を用いて DRAM を接続していた。そこで、
・DRAM と I/O の接続を容易にするため、AX I インタフェースにする。
・手間を減らすため、I/O と VirtI/O はそのまま利用する。
まずは最初からすべての I/O を AXI バスに移行するのではなく、主記憶だけ AXI とする
・また、GUI のブロックデザインでなく、Verilog のテキストコードで移植し、ブロックデザインは、安定稼働してから、対応する。
なお、FiC ボードには、外部 I/O に、電圧レベル変換を行って次の I/O を接続した。
・UART
・SPI-μ SD
・GPIO Ethernet PHY

I/O はそのまま利用したので、Liux のドライバなどは RV-PC をそのまま利用できる。
FiC ボードへの移植手順として、
1)Digilient の Arty, GeneSys などのボード向けに AXI 化してデバッグする
2)SD でなく、UART で OS カーネルと Ram Disk をダウンロードする。
まず、行数の少ない遂次版を AXI 化してから、
パイプライン版を AXI 化し、I/O を接続する。

そこで、まずは、DRAM を AXI 化し、Arty および Genesys2 で動作確認後 FiC へ移植した。

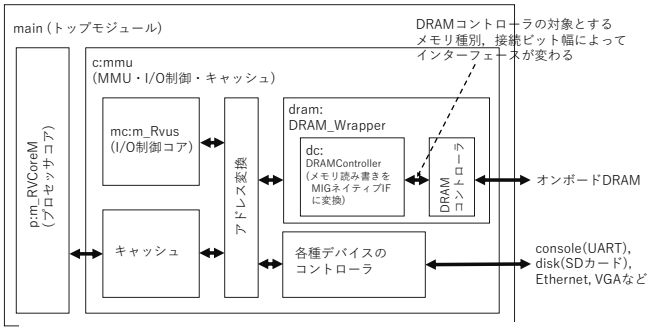

図３．オリジナルの RV-PC のモジュール構成．DRAM コントローラで MIG のネイティブ I/F を採用

DRAM コントローラの対象とするメモリ種別，接続ビット幅によってインタフェースが変る。DRAM コントローラで MIG のネイティブ I/F を採用していたため DRAM の変更(ターゲットボードの変更)時の移植コストが大きいなどの課題があった。図３にオリジナルのモジュール構成を示す。
そこで、次の修正を行った。
・メモリコントローラを移植性向上のため AXI I/F でトップモジュールまで引き出し、DRAM コントローラに接続して、
・メモリコアをトップモジュール直下に配置、ボード毎の構成変更を容易にした。
このような構成とすることで、ターゲットメモリによらず AXI で DRAM コントローラと接続できるため様々なボードへの移植が容易となった。修正後の構成を図４に示す。

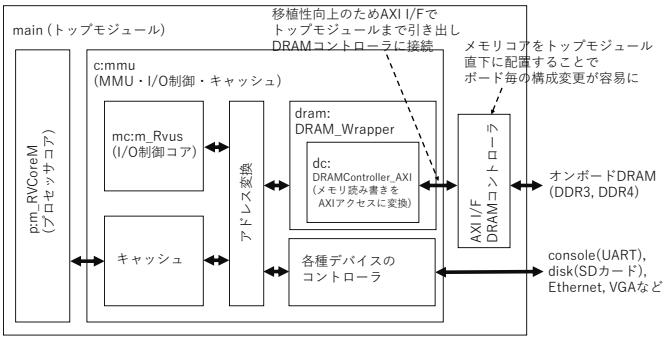

図４。修正後のメモリインターフェースを AXI 化した RV-PC のモジュール構成.

メモリマップされた VirtI/O については、表１に示すようにそのまま利用したので、Linux は何ら改変せず、
OS カーネル、SD カードイメージをそのまま利用できた。
FiC の移植終了後の基板を図５、図６に示す。

表１．メモリマップされた VirtI/O による I/O

デバイス	VirtI/O アドレス
console	0x40000000
disk	0x41000000
ether	0x42000000
keyboard	0x43000000
mouse	0x44000000
frame buffer	0x45000000
0PLIC	0x50000000
CLINT	0x60000000
DRAM	0x80000000

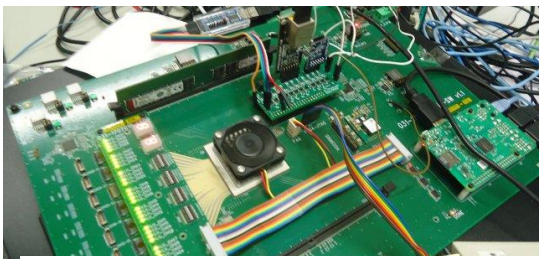

図５　移植後の FiC ボード

６.性能評価

現在次の FPGA ボード

Digilient 社　　Arty A7-100T
　　Nexsys4 DDR
　　GeneSys2
e-trees.Japan 社　exStick GE
PALTEK 社　　FiC

で RV-PC と Linux が稼働している。現在、Xilinx 社 KCU1500 ボードに移植中である。

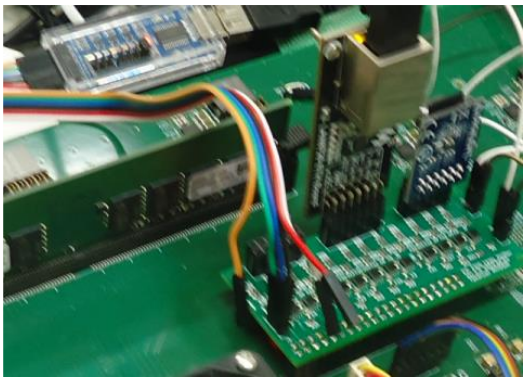

図６.電圧レベル変換基板、USB-UART、GPIO ethernet PHY モジュールカードと

これらすべての FPGA ボードで Linux カーネルは共通、ファイルシステムとブートイメージの書かれた SD カードイメージは共通である。Nexsys4 DDR では Xwindow も使えるが、他は UART シリアル、μSD を Pmod カードで、イーサネットを GPIO 経由の拡張カードで利用できる。

Dhrystone による実行性能を表２に示す。項番３の５ステージパイプライン版は項番５の遂次版の５倍程度の性能となっている。表３にリソース使用量を示す。

表には ASIC チップ化された RISC-V の Linux が稼働する Sifive、組込みの AI アクセラレータに使われる Maix bit 社 K-210、Arm Hardcore の PYNQ-Z1 の ZYNQ、VCU118 だが CVA6 の文献[9]の値も参考までに示す。

表２．Dhrystone による評価

項番	board		core clock(MHz)	device	DRAM	I/O	DMIPS/MHz	DMIPS	Dhrystone/s
1	FiC		75	xcku095ffvb2104-1	DDR4 4GB	ether,uart,sd	0.28	21.32	37453
2	ExstickGE		75	xc7a200tsbg484-2	DDR3 256MB	uart,sd	0.28	21.32	37453
3	ArtyA7-100T (５段パイプライン)		75	xc7a100tcsg324-1	DDR3 256MB	ether,uart,sd, vga,kb,mouse	0.29	21.56	37878
4	Arty(遂次)		100	xc7a100tcsg324-1	DDR3 256MB	uart	0.04	4.33	7616
5	Nexsys4DDR		75	xc7a100tcsg324-1	DDR2 128MB	ether,uart,sd, vga,kb,mouse			
6	genesys2		75	xc7k325tffg900-2	DDR3 256MB	ether,uart,sd	0.28	20.85	36630
7	CVA6[9]			VCU118			1.21		
8	Sifive	ASIC	1200				1.84	2205.17	3874484
9	MaixBit/k210	ASIC	400				0.70	279.64	491328
10	PynqZ1	ASIC:hardcor	650				1.12	725.03	1273885

表３．FPGA のリソース使用量

項番	board	device	DRAM	I/O	CLB LUTs	CLB Regs	BRAM	DSP
1	FiC	xcku095ffvb2104-1	DDR4 4GB	ether,uart,sd	6.10(32160/537600)	2.82(30342/10752	3.72(62.5/1680)	2.34(18/768)
					Slice LUTs	Slice Regs		
2	ExstickGE	xc7a200tsbg484-2	DDR3 256MB	uart,sd	10.96(14752/134600)	3.49(9406/269200)	11.51(42/365)	2.03(15/740)
3	ArtyA7-100T (５段パイプライン)	xc7a100tcsg324-1	DDR3 256MB	ether,uart,sd, vga,kb,mouse	24.07(15262/63400)	7.75(9825/126800)	97.78(132/135)	6.67(16/240)
4	Arty(遂次)	xc7a100tcsg324-1	DDR3 256MB	uart	11.43(7249/63400)	2.79(3535/126800)	28.15(38/135)	6.25(15/240)
5	Nexsys4DDR	xc7a100tcsg324-1	DDR2 128MB	ether,uart,sd, vga,kb,mouse	18.60(11790/63400)	4.87(6179/12800)	98.52(133/135)	6.67(16/240)
6	genesys2	xc7k325tffg900-2	DDR3 256MB	ether,uart,sd	8.92(18179/203800)	3.12(12707/40760	9.44(42/445)	1.79(15/840)

なお、AXI 化したコードなどは、それぞれ次の gitub から利用できる。

1)起動イメージを UART からアップロードする版(表 2,3 の項番 4 の RISC-V、RVSoC) :

https://github.com/miyo/rvsoc_axi

2)SD カードから Linux ブート, Ethernet が利用できる Linux 版(表２， ３の項番３の RISC-V、RV-PC) :

https://github.com/miyo/rv-pc-axi

７．おわりに

AXI バスの導入などにより、FiC FPGA ボードに Linux が動く RISC-V を移植し、独自ハードウェアを追加可能なシステムソフトウェアの研究環境を実現した。

今後の課題は

・性能チューニング

・高速リンクの実現

・Alveo などの PCIe スロットでアクセラレータとして使用できる FPGA ボードへの移植(KCU1500 は作業中)

・Vivado のブロックデザイン化：IP ライブラリ化

・ASIC 化

となっており、 CREST プロジェクトでは、本 RISC-V の ASIC 化も検討している。

謝辞

RV-PC の使用を快く快諾いただき、技術情報をご提供いただいた東京工業大学 情報理工学院 情報工学系 吉瀬 謙二教授に深く感謝する。

また、本研究は JST CREST Society 5.0 を支える革新的コンピューティング技術「MEC 用マルチノード統合システムの開発」(JPMJCR19K1)の支援を受けた。

参考文献

[1] David Patterson, and Andrew Waterman: The RISC-V Reader: AnOpen Architecture Atlas, Strawberry Canyon, ISBN/0999249118 , 2017.

[2] Junya Miura, Hiromu Miyazaki, Kenji Kise:A portable and Linux capable RISC-V computer system in Verilog HDL arXiv:2002.03576 [cs.AR] 2020-02.

[3] https://www.arch.cs.titech.ac.jp/wk/rvsoc/doku.php

[4] iromu MIYAZAKI, Takuto KANAMORI, Md Ashraful ISLAM, Kenji KISE, RVCoreP: An Optimized RISC-V Soft Processor of Five-Stage Pipelining, IEICE Transactions on Information and Systems, 2020, Volume E103.D, Issue 12, Pages 2494-2503, 2020.12

[5]https://www.arch.cs.titech.ac.jp/wk/rvcore/doku.php

[6] 吉瀬謙二：FPGA 入門用ボードで手軽に実現する Linux 対応の RISC-V パーソナルコンピュータ,-RISC-V Days Spring,-RISC-V 協会,
https://riscv.or.jp/wp-content/uploads/2021-04-23-kise_compressed.pdf, 2022-04-23.

[7] Sun, Y., Amano, H: FiC-RNN: A multi-FPGA acceleration framework for deep recurrent neural networks, IEICE Transactions on Information and SystemsVolume E103D, Issue 12, 1 December 2020, pp.2457-2462, 2020.12

[8] 塚健介・天野英晴: AlexNet のマルチ FPGA システムへの分割検討と実装, 信学技報, vol. 119, no. 147, CPSY2019-23, pp. 91-95, 2019.7.

[9] .Dorflinger , M.Albers , B.Kleinbeck , Y.Guan , H.Michalik A comparative survey of opensource application-class RISC-V processor implementations, CF '21: Proceedings of the 18th ACM International Conference on Computing Frontiers, pp.12–20, 2021.3

図 7 .Linux ブート時のコンソール画面