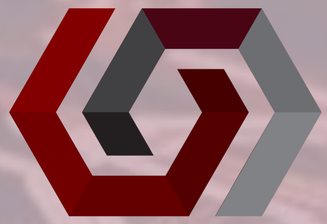


OPEN-SOURCE CPU: RISC-V CFU AND ZEPHYR

Mohammed Billoo

MAB Labs Embedded Solutions

FOSDEM 25

Mohammed Billoo

(mab@mab-labs.com)

THE SPEAKER

/mab-embedded

@mabembedded

- Embedded Software Consultant (NYC, USA)
- Design Work
  - Medical Devices
  - Scientific Instruments
  - LIDAR
  - Custom ASICs
- Experience/Expertise
  - Zephyr RTOS
  - Embedded Linux
  - GUI-based applications

BIOS FOOD NEWSLETTER

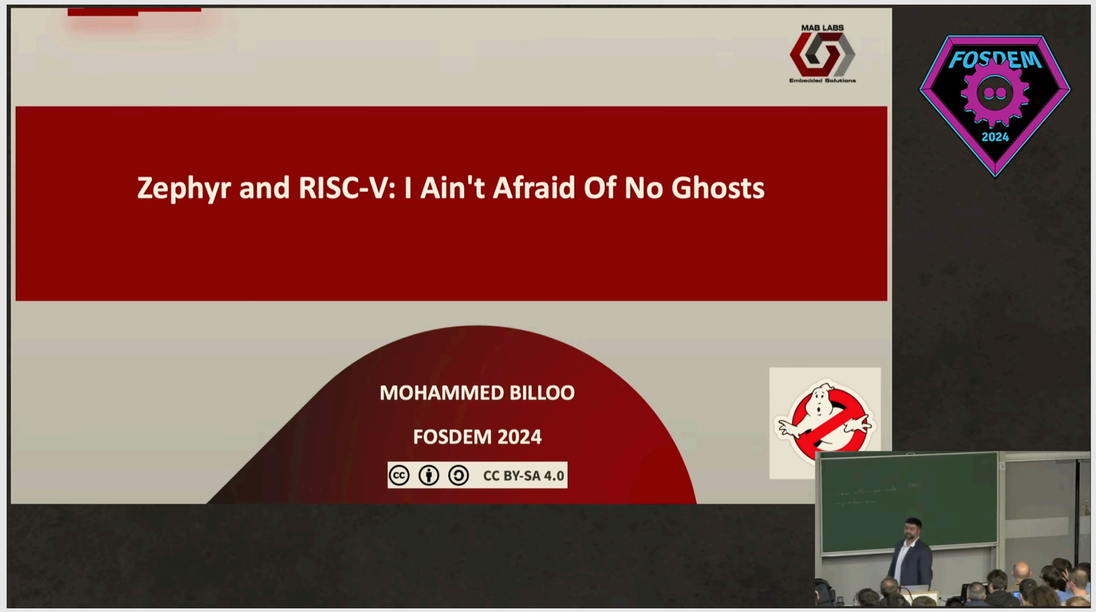
Training/Workshops

www.mab-labs.com

AGENDA

- Background
- Under the Hood
- The Plan
- How It Went (not good)
- Next Steps (need debugging help)

BACKGROUND



https://www.youtube.com/watch?v=syA3xxKfB4s&t=28s

- FOSDEM24
- Used neorv32 project to implement RISC-V ISA in an FPGA
  - Has support for built-in bootloader
- Used vendor tools to build and flash FPGA
- Loaded “Hello World” Zephyr application on to FPGA.
  - Zephyr has support for “neorv32” board
  - Uploaded application using bootloader

BACKGROUND

- After some fennagling, got output!
- See previous presentation for details
- Next step was to investigate CFU
- Roadblock
  - Design with CFU enabled didn’t fit in FPGA

```
CMD:> h
Available CMDs:
 h: Help
 r: Restart
 u: Upload
 s: Store to flash
 l: Load from flash
 x: Boot from flash (XIP)
 e: Execute
CMD:> e
Booting from 0x00000000...

*** Booting Zephyr OS build 6f56a6a91e2c ***
Hello World! neorv32
```

MOTIVATION

- **C**ustom **F**unction **U**nit?
- Allows us to offload functionality to hardware (FPGA)
- Specific for operations that are inefficient (in software):
  - Performance
  - Latency
  - Energy Consumption
  - Program Memory

MOTIVATION

- AI
- Crypto
- Communications
- Arithmetic
- Image Processing
- **Requires CPU dependency**

DETAILS

- Leverages “custom-0” and “custom-1” RISC-V opcodes
- CFU is implemented using the “**Zxcfu**” extension of the RISC-V ISA
  - **Specific to neorv32**
- Sample implementation of XTEA in RTL!
  - Simple block cipher
  - Embedded systems

inst[4:2] inst[6:5]	000	001	010	011	100	101	110	111 (> 32*b*)
00	LOAD	LOAD-FP	*custom-0*	MISC-MEM	OP-IMM	AUIPC	OP-IMM-32	48*b*
01	STORE	STORE-FP	*custom-1*	AMO	OP	LUI	OP-32	64*b*
10	MADD	MSUB	NMSUB	NMADD	OP-FP	*reserved*	*custom-2/rv128*	48*b*
11	BRANCH	JALR	*reserved*	JAL	SYSTEM	*reserved*	*custom-3/rv128*	≥ 80*b*

Table 19.1: RISC-V base opcode map, inst[1:0]=11

UNDER THE HOOD

rtl/core/neorv32_cpu_cp_cfu.vhd

```
-- instruction identifiers (funct3 bit-field) --
constant xtea_enc_v0_c : std_ulogic_vector(2 downto 0) := "000";
constant xtea_enc_v1_c : std_ulogic_vector(2 downto 0) := "001";
constant xtea_dec_v0_c : std_ulogic_vector(2 downto 0) := "010";
constant xtea_dec_v1_c : std_ulogic_vector(2 downto 0) := "011";
constant xtea_init_c   : std_ulogic_vector(2 downto 0) := "100";
```

Use “intrinsics” to “call” functions in CFU

sw/example/demo_cfu/main.c

```
#define xtea_hw_init(sum)           neorv32_cfu_r3_instr(0b0000000, 0b100, sum, 0 )
#define xtea_hw_enc_v0_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b000, v0,  v1)
#define xtea_hw_enc_v1_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b001, v0,  v1)
#define xtea_hw_dec_v0_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b010, v0,  v1)
#define xtea_hw_dec_v1_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b011, v0,  v1)
#define xtea_hw_illegal_inst()      neorv32_cfu_r3_instr(0b0000000, 0b111, 0,   0 )
```

UNDER THE HOOD

```
#define xtea_hw_init(sum)           neorv32_cfu_r3_instr(0b0000000, 0b100, sum, 0 )
#define xtea_hw_enc_v0_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b000, v0,  v1)
#define xtea_hw_enc_v1_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b001, v0,  v1)
#define xtea_hw_dec_v0_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b010, v0,  v1)
#define xtea_hw_dec_v1_step(v0, v1) neorv32_cfu_r3_instr(0b0000000, 0b011, v0,  v1)
#define xtea_hw_illegal_inst()      neorv32_cfu_r3_instr(0b0000000, 0b111, 0,   0 )
```

sw/lib/include/neorv32_cpu_cfu.h

```
#define neorv32_cfu_r3_instr(funct7, funct3, rs1, rs2) \
        CUSTOM_INSTR_R3_TYPE(funct7, rs2, rs1, funct3, 0b0001011)
```

UNDER THE HOOD

```
#define neorv32_cfu_r3_instr(funct7, funct3, rs1, rs2) \
        CUSTOM_INSTR_R3_TYPE(funct7, rs2, rs1, funct3, 0b0001011)
```

sw/lib/include/neorv32_intrinsics.h

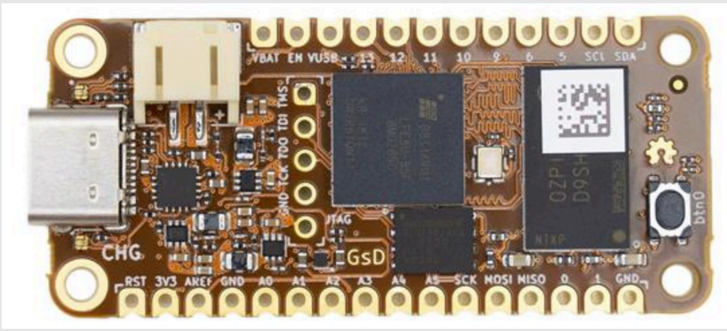
```
#define CUSTOM_INSTR_R3_TYPE(funct7, rs2, rs1, funct3, opcode) \
({                                                             \
    uint32_t __return;                                         \
    asm volatile (                                             \
      ".word (                                                 \
        ((" #funct7 ") & 0x7f) << 25) |                        \
        (((  reg_%2   ) & 0x1f) << 20) |                       \
        (((  reg_%1   ) & 0x1f) << 15) |                       \
        ((" #funct3 ") & 0x07) << 12) |                        \
        (((  reg_%0   ) & 0x1f) <<  7) |                       \
        ((" #opcode ") & 0x7f) <<  0)                          \
      );"                                                      \
      : [rd] "=r" (__return)                                   \
      : "r" (rs1),                                             \
        "r" (rs2)                                              \
    );                                                         \
    __return;                                                  \
})
```

Assembler calls!

THE PLAN

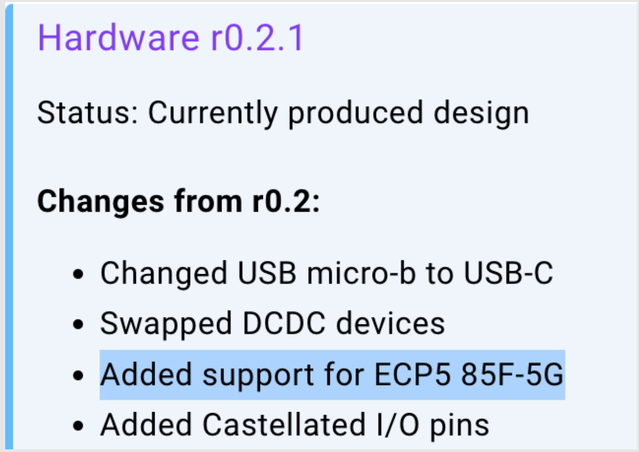
- Get board with FPGA that can fit design with CFU
- Enable CFU in board top level RTL file
- Build and flash design
- Load Zephyr Blinky/Hello World
- Add function calls to Zephyr from neorv32 library to call instrinsics
  - Need to pull in header/source files from neorv32 repo
- Confirm XTEA operation
  - Compare performance of XTEA in HW vs SW

https://orangecrab-fpga.github.io/orangecrab-hardware/



HOW IT WENT

- OrangeCrab
  - Need newer hardware version
  - Larger FPGA to support CFU functionality
- https://github.com/stnolting/neorv32-setups/tree/main/osflow
  - Describes process to build FPGA image
  - OrangeCrab “setup” uses open-source tools
  - Current implementation uses smaller FPGA
    - Lattice ECP5-25F - requires modifications



Hardware r0.2.1

Status: Currently produced design

Changes from r0.2:

- Changed USB micro-b to USB-C
- Swapped DCDC devices
- Added support for ECP5 85F-5G
- Added Castellated I/O pins

Orange_Crab	GHDL, Yosys, nextPNR	Orange Crab	Lattice ECP5-25F	umarcor, jeremyherbert

HOW IT WENT

- Installing tools directly didn’t go as planned - ran into installation issues
- Decided to leverage container that is used as part of Github Actions build in repo

```
FROM gcr.io/hdl-containers/debian/bullseye/impl

RUN apt-get update -qq \
 && DEBIAN_FRONTEND=noninteractive apt-get -y install --no-install-recommends \
  python3-pip \
 && pip3 install wheel setuptools \
 && pip3 install doit \
 && apt-get autoclean && apt-get clean && apt-get -y autoremove \
 && rm -rf /var/lib/apt/lists/*

ENV GHDL_PLUGIN_MODULE=ghdl

WORKDIR tmp/src
```

```
[~]$ cd fosdem/2025/neorv32-setups
[neorv32-setups]$ ls
CODE_OF_CONDUCT.md  constraints  LICENSE  osflow   radiant    vivado
cologne_chip        gowineda     neorv32  quartus  README.md
[neorv32-setups]$ docker run -v$PWD:/tmp/src:z -it orangecrab-neorv32 bash -c 'make -C osflow/ BOARD
=OrangeCrab MinimalBoot'
```

HOW IT WENT

- Enable DFU in RTL

```
[neorv32]$ git diff
diff --git a/rtl/core/neorv32_top.vhd b/rtl/core/neorv32_top.vhd
index 1236ed22..6b5150cf 100644
--- a/rtl/core/neorv32_top.vhd
+++ b/rtl/core/neorv32_top.vhd
@@ -45,7 +45,7 @@ entity neorv32_top is
     CPU_EXTENSION_RISCV_Zicond : boolean                          := false;       -- implement integer conditional operations?
     CPU_EXTENSION_RISCV_Zihpm  : boolean                          := false;       -- implement hardware performance monitors?
     CPU_EXTENSION_RISCV_Zmmul  : boolean                          := false;       -- implement multiply-only M sub-extension?
-    CPU_EXTENSION_RISCV_Zxcfu  : boolean                          := false;       -- implement custom (instr.) functions unit?
+    CPU_EXTENSION_RISCV_Zxcfu  : boolean                          := true;       -- implement custom (instr.) functions unit?
```

HOW IT WENT

- Modify appropriate Makefile to inform tools of different FPGA

```
diff --git a/osflow/boards/index.mk b/osflow/boards/index.mk
index 20ff948..fd09c43 100644
--- a/osflow/boards/index.mk
+++ b/osflow/boards/index.mk
@@ -99,7 +99,7 @@ DEVICE_SERIES = ecp5
 OrangeCrab_REV ?= r02-25F

 CONSTRAINTS ?= $(PCF_PATH)/$(BOARD).lpf
-PNRFLAGS    ?= --25k --package CSFBGA285 --ignore-loops --timing-allow-fail
+PNRFLAGS    ?= --85k --package CSFBGA285 --ignore-loops --timing-allow-fail
 IMPL        ?= neorv32_$(BOARD)_$(OrangeCrab_REV)_$(ID)

 endif
```

Ideally should update this, but used elsewhere and resulted in build failures.

HOW IT WENT

- After successfully built FPGA, used `dfu-util` to Flash FPGA
  - Held button on power up to enter programming mode
- Attach device information to file
  - `cp neorv32_OrangeCrab_r02-25F_MinimalBoot.bit neorv32_OrangeCrab_r02-25F_MinimalBoot.dfu`
  - `dfu-suffix -v 1209 -p 5af0 -a neorv32_OrangeCrab_r02-25F_MinimalBoot.dfu`
  - `dfu-util -a 0 -D neorv32_OrangeCrab_r02-25F_MinimalBoot.dfu`

HOW IT WENT

- Bootloader working!
- Compile Zephyr “Hello World” for RISC-V
  - `west build -p -b neorv32 samples/hello_world/`
- On power-up, hit any key over console to enter bootloader
- Use neorv32 script to upload binary
- Run into issue when attempting to flash application
  - https://stnolting.github.io/neorv32/

`ERR_SIZE`	Your program is way too big for the internal processor’s instructions memory. Increase the memory size or reduce your application code.

```
<< NEORV32 Bootloader >>

BLDV: Jul 19 2024
HWV:  0x01100209
CLK:  0x016e3600
MISA: 0x40800100
XISA: 0x0000008b
SOC:  0x0013000d
IMEM: 0x00004000
DMEM: 0x00002000

Autoboot in 10s. Press any key to abort.
Loading from SPI flash @0x00400000...

ERR_SIZE
```

HOW IT WENT

- Increase IMEM size

```
index 552a744..ed13551 100644
--- a/osflow/board_tops/neorv32_OrangeCrab_BoardTop_MinimalBoot.vhd
+++ b/osflow/board_tops/neorv32_OrangeCrab_BoardTop_MinimalBoot.vhd
@@ -102,7 +102,7 @@ begin
   neorv32_inst: entity work.neorv32_ProcessorTop_MinimalBoot
   generic map (
     CLOCK_FREQUENCY => f_clock_c,  -- clock frequency of clk_i in Hz
-    MEM_INT_IMEM_SIZE => 16*1024,
+    MEM_INT_IMEM_SIZE => 32*1024,
     MEM_INT_DMEM_SIZE => 8*1024
   )
   port map (
```

HOW IT WENT

- After re-building FPGA and re-uploading application, no “Hello World” :-(
- Try demo Blinky example under `neorv32/sw`
- Same issue
  - No indication of error
  - Console just sits there

NEXT STEPS

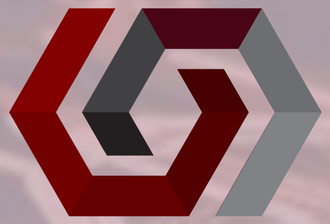
- Troubleshoot why application not loading
- What debugging tools available and how to debug?
  - Equivalent of JTAG/SWD and GDB in FPGA land?
  - Have used Xilinx ILA and Synopsys Identify
    - How to capture signals in Lattice?
    - Open source tools?
    - Necessary hardware connections to perform troubleshooting?

NEXT STEPS

- Get it working and generate comparison metrics in Zephyr
  - Present success at FOSDEM26 ?
- Integrate CFU intrinsic in Zephyr and upstream
- Investigate Custom Functions Subsystem
  - CPU independent operations

NEXT STEPS

- Try other hardware-accelerated functions
- CFU Playground
  - https://cfu-playground.readthedocs.io/en/latest/
  - Different design/implementation
  - Meant for machine learning
  - Integrate with Zephyr and present results!
- Custom Functions Subsystem
  - CPU independent
  - Integrate with Zephyr and present results!



THANK YOU!

Mohammed Billoo

MAB Labs Embedded Solutions

FOSDEM25