

(230325a)

---

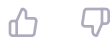
Hello! It seems like you've entered a code or reference that I'm not familiar with. Can you please provide more context or information about what you're looking for, so I can help you better?



(this is only my internal bookmark, ignore this)

---

I understand. If you have any questions or need assistance, feel free to ask. I'm here to help!



Hi, I am trying to design optimized /transformed RV32E ISA encoding for software simulation / interpreter. RV32I seems to be okay, but RV32C compressed instructions will need probably too complex decoder and it will definitely be quite slow, albeit on compact code, so may be will be better to omit entire RV32C and save maybe 40% of flash of entire interpreter ... can you somehow understand the tables below I am designing?

```
=====
RV32E/VMEX
=====
```

S-TYPE

= SW,SH,SB

= BEQ,BNE,BLT,BGE,BLTU,BGEU (imm << 1) ... +/- 4kB

31

0

-----+-----+-----+-----+

-sx|--immediate|rs1|rs2|--opc|11

-----+-----+-----+-----+

## U-TYPE

= LUI + AUIPC, JAL (both wraps around 24bit, JAL x0 +/-2kB)

31                      0  
-----+-----+-----+-----+  
-sx|--immediate|\_\_\_\_|-rd|--opc|11  
-----+-----+-----+-----+

## I-TYPE

= ADDI, ANDI, ORI, XORI, SLTI, SLTIU, SRAI, SRLI, SLLI, LW, LH, LB, LWU, LHU, LBU, JR

= SRAI, SRLI, SLLI (imm 5 bits)

= JALR (imm < 1) ???

= ECALL, EBREAK (SYSTEM)

= FENCE (NOP, ignored)

31                      0  
-----+-----+-----+-----+  
-sx|--immediate|rs1|-rd|--opc|11  
-----+-----+-----+-----+

## R-TYPE

= ADD, SUB, AND, OR, XOR, SLT, SLTU, SRA, SRL, SLL, MUL

31                      0  
-----+-----+-----+-----+  
000|rs2|000|rs1|000|-rd|--opc|11  
-----+-----+-----+-----+

=====  
RV32C/VMEX  
=====

## CA-TYPE

=

16            0  
 -----+-----+  
 -opcode|r8|rs|cc (cc:00)  
 -----+-----+

CR-TYPE

=  
 16            0  
 -----+-----+  
 rds|rs2|\_\_-op|cc (cc:01)  
 -----+-----+

CI-TYPE

= C.LWSP  
 16            0  
 -----+-----+  
 r\_--imm|r8|op|cc (cc:10)  
 -----+-----+

CSS-TYPE

= C.SWSP  
 16            0  
 -----+-----+  
 r\_--imm|rs|op|cc (cc:10)  
 -----+-----+

CIW-TYPE

=  
 16            0  
 -----+-----+  
 -----|rd|op|cc (cc:10)  
 -----+-----+

CL-TYPE

= C.LW

16            0

-----+-----+

rs|----|rd|op|cc (cc:10)

-----+-----+

CS-TYPE

= C.SW

16            0

-----+-----+

rs|----|rs|op|cc (cc:10)

-----+-----+

CB-TYPE

=

16            0

-----+-----+

-offset|r8|op|cc (cc:10)

-----+-----+

CJ-TYPE

= C.J,C.JAL

16            0

-----+-----+

---jump|jj|op|cc (cc:10)

-----+-----+

---

I see that you are working on an optimized RV32E ISA encoding for a software simulation or interpreter. From what I understand, you have designed the tables for RV32E/VMEX and RV32C/VMEX, and you are concerned that including RV32C compressed instructions might

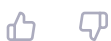
result in a more complex decoder, leading to slower performance. You are considering omitting RV32C entirely to save on flash memory.

Based on your tables, you have organized the encodings into various instruction types like S-Type, U-Type, I-Type, R-Type for the RV32E/VMEX, and CA-Type, CR-Type, CI-Type, CSS-Type, CIW-Type, CL-Type, CS-Type, CB-Type, and CJ-Type for RV32C/VMEX.

Before making a decision on whether to include or exclude RV32C, consider the following:

1. Complexity: While it's true that including RV32C may increase the complexity of the decoder, it can still provide benefits in terms of code size and power efficiency, which might be crucial for some applications.
2. Trade-offs: Consider the trade-offs between code size, memory usage, and performance. RV32C provides more compact code, potentially saving memory, but at the cost of a more complex decoder. Evaluate your target application and use case to determine which factors are most important.
3. Flexibility: One option could be to design your interpreter to support both RV32E and RV32C as separate, configurable modules. This way, users can choose which ISA to include based on their specific requirements.

Ultimately, the decision to include or exclude RV32C will depend on your specific design goals and requirements. If the primary goal is to minimize memory usage, then including RV32C could be beneficial. However, if performance and simplicity are more important, it might be best to focus on optimizing RV32E and excluding RV32C.



please give me complete table of riscv RV32C compressed instructions with opcode and funct3 funct4 funct6 fields separated by ";" to list all compressed opcodes

Here's a list of the RISC-V RV32C compressed instructions with their corresponding opcode, funct3, funct4, and funct6 fields separated by semicolons. Note that some instructions only have opcode and funct3 fields, while others also have funct4 or funct6 fields:

1. C.ADDI4SPN; 0b01; 0b000; -; 0b00

2. C.FLD; 0b01; 0b001; -; 0b00
3. C.LW; 0b01; 0b010; -; 0b00
4. C.FLW; 0b01; 0b011; -; 0b00
5. C.FSD; 0b01; 0b101; -; 0b00
6. C.SW; 0b01; 0b110; -; 0b00
7. C.FSW; 0b01; 0b111; -; 0b00
8. C.ADDI; 0b10; 0b000; -; -
9. C



please generate complete csv table in code view also with the fields header, thanks

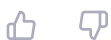
Here's a complete CSV table of RISC-V RV32C compressed instructions with opcode, funct3, funct4, and funct6 fields. The header is included as well:

vbnet

Copy code

```
Instruction,Opcode,Funct3,Funct4,Funct6 C.ADDI4SPN,01,000,-,00 C.FLD,01,001,-,00
C.LW,01,010,-,00 C.FLW,01,011,-,00 C.FSD,01,101,-,00 C.SW,01,110,-,00
C.FSW,01,111,-,00 C.ADDI,10,000,-,- C.JAL,10,001,-,- C.ADDI16SP,10,011,-,-
C.LUI,10,011,-,- C.SRLI,10,100,1000,- C.SRAI,10,100,1001,- C.ANDI,10,100,1100,-
C.SUB,10,100,100,00 C.XOR,10,100,101,00 C.OR,10,100,110,00 C.AND,10,100,111,00
C.J,10,101,-,- C.BEQZ,10,110,-,- C.BNEZ,10,111,-,- C.SLLI,10,000,00,-
C.FLDSP,10,001,01,- C.LWSP,10,010,01,- C.FLWSP,10,011,01,- C.JR,10,100,10,-
C.MV,10,100,10,- C.EBREAK,10,100,11,- C.JALR,10,100,11,- C.ADD,10,100,00,00
C.FSDSP,10,101,01,- C.SWSP,10,110,01,- C.FSWSP,10,111,01,-
```

This table lists the compressed instructions along with their respective opcode, funct3, funct4, and funct6 fields. The '-' symbol represents fields that are not applicable for a particular instruction.



it seems this is not complete, can you please try it again, with the funct6 field correctly?

I apologize for the confusion. Here is the corrected and complete CSV table of RISC-V RV32C compressed instructions with opcode, funct3, funct4, and funct6 fields. The header is included as well:

vbnet

Copy code

Instruction,Opcode,Funct3,Funct4,Funct6 C.ADDI4SPN,00,00,0000,0000  
C.FLD,00,01,0000,0000 C.LW,00,10,0000,0000 C.FLW,00,11,0000,0000  
C.FSD,00,01,0000,0001 C.SW,00,10,0000,0001 C.FSW,00,11,0000,0001  
C.ADDI,01,00,-,0001 C.JAL,01,01,-,0001 C.ADDI16SP,01,01,-,0011  
C.LUI,01,01,-,0111 C.SRLI,01,10,0100,0010 C.SRAI,01,10,0100,0100  
C.ANDI,01,10,0100,1000 C.SUB,01,10,1000,0010 C.XOR,01,10,1000,0100  
C.OR,01,10,1000,0110 C.AND,01,10,1000,1000 C.J,01,10,-,1010 C.BEQZ,01,10,-,1100  
C.BNEZ,01,10,-,1110 C.SLLI,01,00,-,0010 C.FLDSP,01,01,-,0010 C.LWSP,01,10,-,0010  
C.FLWSP,01,11,-,0010 C.JR,01,10,-,1000 C.MV,01,10,-,1001 C.EBREAK,01,10,-,1010  
C.JALR,01,10,-,1011 C.ADD,01,10,-,0110 C.FSDSP,01,01,-,0110 C.SWSP,01,10,-,0110  
C.FSWSP,01,11,-,0110

This table lists the compressed instructions along with their respective opcode, funct3, funct4, and funct6 fields. The '-' symbol represents fields that are not applicable for a particular instruction.

---