Adding Opcodes to RISCV-Toolchain

Basic Introduction

Pascal Pieper

Pascal.Pieper@dfki.de

Deutsches Forschungsinstitut
für Künstliche Intelligenz

Cyber-Physical Systems

Overview

- Choose a fitting instruction format (see fig. 1)
- Add custom instruction name and bitformat to `custom-opcodes`
- Insert generated bitmasks into `riscv-opc.h`
- Add assembler commands to `riscv-opc.c`

Instruction Types

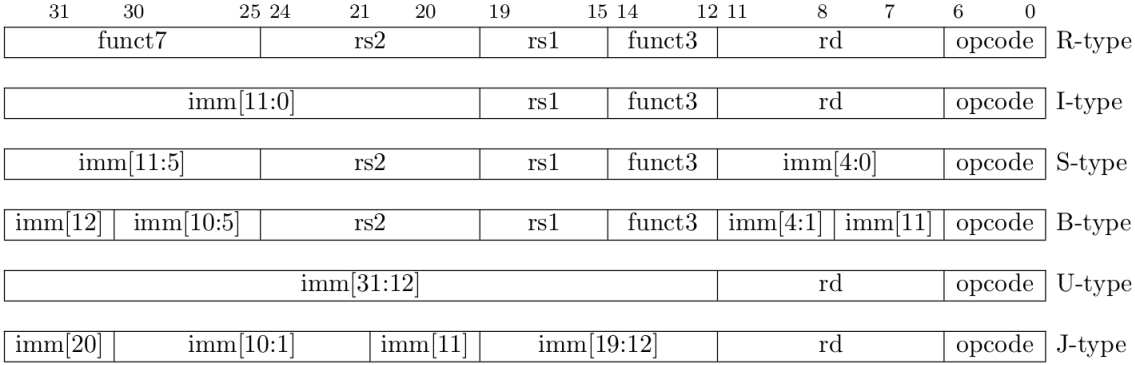

Figure: RISC-V base instruction formats showing immediate variants

Custom Opcode

`/riscv-opcodes/opcodes-custom`

- choose applicable opcode (6..2, see fig. 2)
- Add custom instruction name and format to `custom-opcodes`

→ Operation code ≠ Assembler instruction!

File *custom-opcodes:*

```
# Operation    rd source1  source2   func3    code      non-compressed
settaint.i     rd 19..15=0 imm12    14..12=0 6..2=0x0A 1..0=3  #I Type
settaint.r     rd rs1      31..20=0 14..12=1 6..2=0x0A 1..0=3  #R-Type
gettaint       rd rs1      31..20=0 14..12=2 6..2=0x0A 1..0=3  #R-Type
```

Opcodes

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

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

Custom Opcode

/riscv-opcodes/

- Run mask generator:

```
$ cat opcodes-custom | ./parse-opcodes
```

Assembler instruction

/riscv-binutils/include/opcode/riscv-opc.h

- Insert generated mask and match bitmasks

File *riscv-opc.h:*

```
...
#define MATCH_SETTAINT_I 0x002b
#define MASK_SETTAINT_I  0xff07f
#define MATCH_SETTAINT_R 0x102b
#define MASK_SETTAINT_R  0xfff0707f
#define MATCH_GETTAINT   0x202b
#define MASK_GETTAINT    0xfff0707f
...
```

Assembler instructions

/riscv-binutils/opcodes/riscv-opc.c

- Add custom assembler instructions

File *riscv-opc.c:*

```
...
{"add",         0, {"I", 0},   "d,s,j",  MATCH_ADDI, MASK_ADDI, match_opcode, INSN_ALIAS },
//CUSTOM
{"settaint",    0, {"I", 0},   "d,j", MATCH_SETTAINT_I, MASK_SETTAINT_I, match_opcode, 0 },
{"settaint",    0, {"I", 0},   "d,s", MATCH_SETTAINT_R, MASK_SETTAINT_R, match_opcode, 0 },
{"gettaint",    0, {"I", 0},   "d,s", MATCH_GETTAINT  , MASK_GETTAINT  , match_opcode, 0 },

{"la",          0, {"I", 0},   "d,B",  0,    (int) M_LA,  match_never, INSN_MACRO },
...
```

Build

/

- Configure and Build!
- Verify by using the new Instruction in inline assembly
  - `asm volatile ( "gettaint %[x], %[y]" : [x] "=r" (taintval) : [y] "r" (*word) );`
  - `riscv/bin/riscv32-unknown-elf-gcc -g test.c -o test`
  - `riscv/bin/riscv32-unknown-elf-objdump -dS test`

```
...
1021c:       0007c783                lbu     a5,0(a5)
10220:       0007a7ab                gettaint        a5,a5
10224:       fef407a3                sb      a5,-17(s0)
...
```