



Virtual Platforms for early Embedded Software Development

RISC-V 8th Workshop – Barcelona

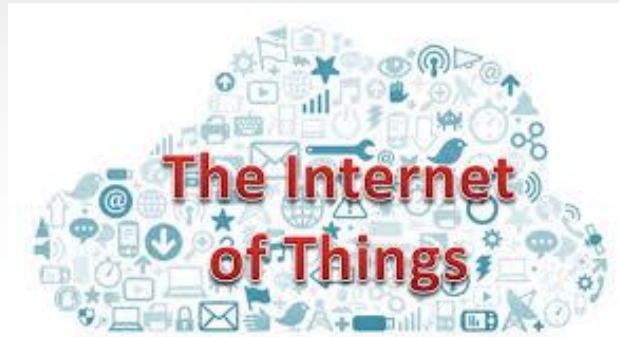
Wednesday May 09, 4:00pm

Kevin McDermott & Lee Moore – Imperas Software

Hugh O'Keeffe – Ashling

New Markets With New Software Requirements

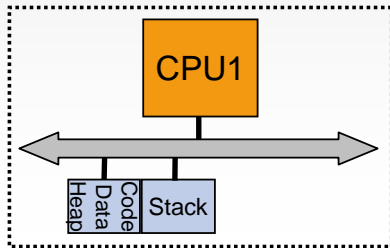
- Schedule
- Quality
- Reliability
- Security
- Safety
- Engineering productivity / automation
- Predictability on software development schedules
- Unknown / unmeasurable software delivery risk



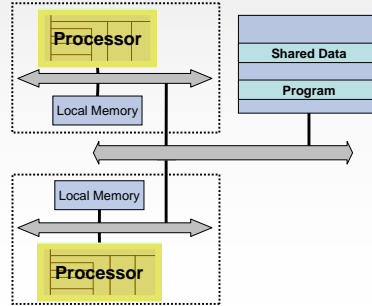
Virtual Platforms Accelerate Software Development

- A key to the adoption of RISC-V is Software
 - Need processors/platforms for software development
 - Start porting existing software to RISC-V
- Virtual platforms (software simulation)
 - Available months before hardware
 - Can be used in hardware verification
 - Can accelerate software porting and development
 - Can help bring up of software on new platforms
 - Support of RISC-V instruction extensions

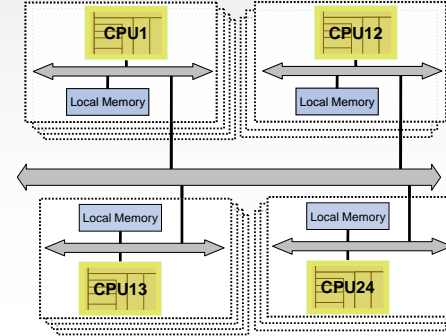
Processor Platform Configurations



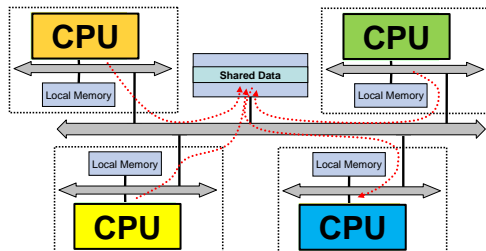
Single core, simple



Multi-core shared memory

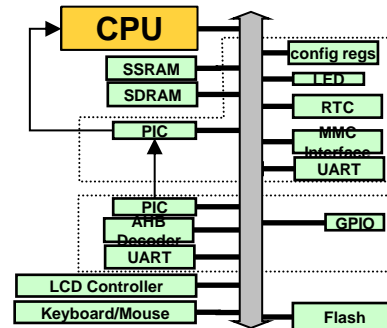


Many-cores

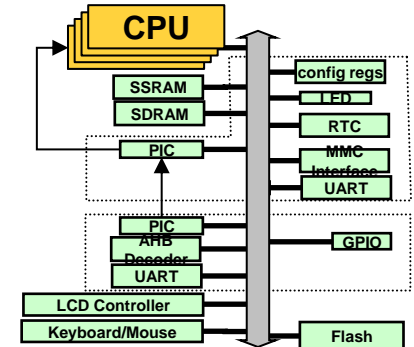


Heterogeneous

Single Core



Multi Core (SMP)



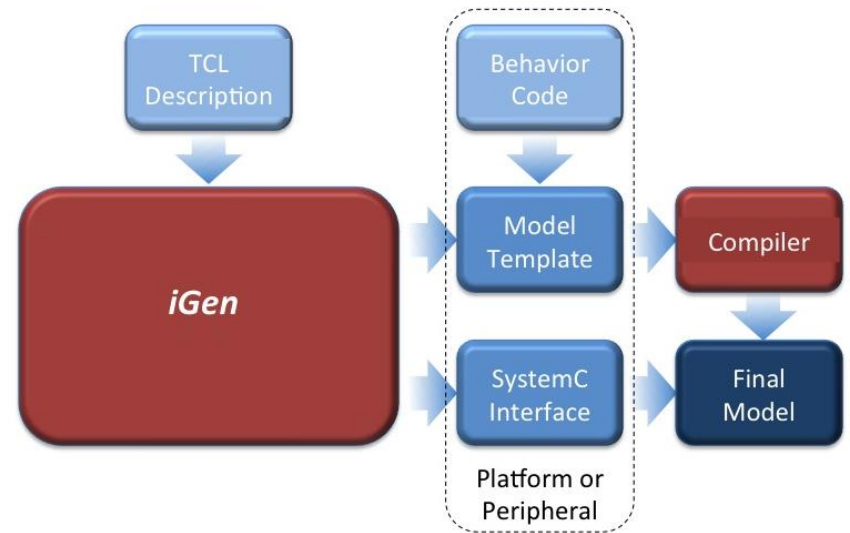
Booting OS, eg Linux

Extendable Platform Kits™ (EPKs™)

- EPKs are virtual platforms
 - with software set-up, help users to start quickly
- EPKs include
 - Individual models, binary and source
 - Platform model, binary and source
 - Software and/or OS running on platform

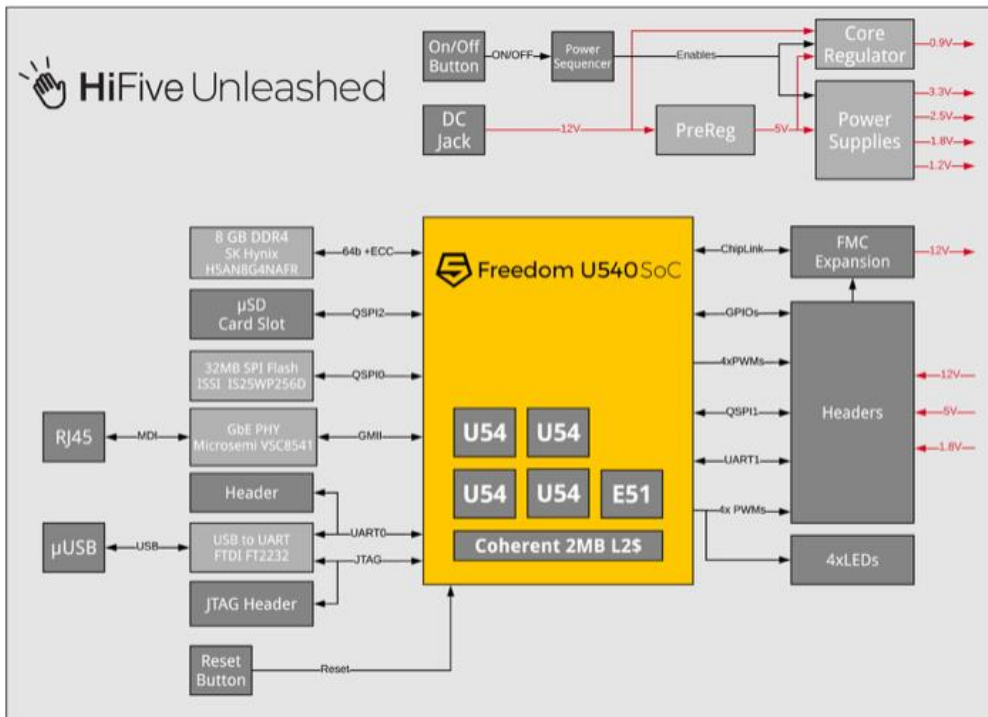
- 200+ Processor Models
- 50+ EPKs
- 100s of peripheral models available in the OVP Library
- All models are open source
 - Distributed under the Apache 2.0 open source license
- All models have both C and SystemC interfaces

- Peripherals: users define pins and registers, and functionality
- Platforms: users define memory, component connectivity



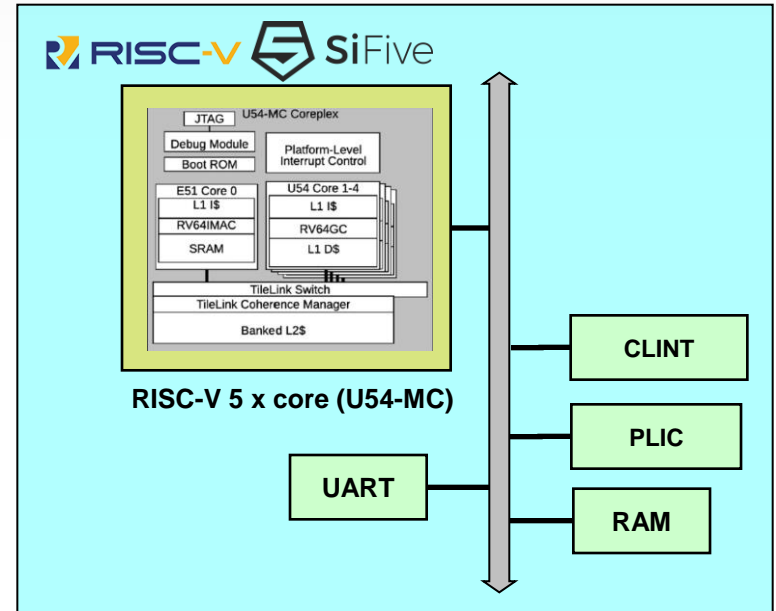
RISC-V EPK based on SiFive U54-MC

The Virtual Platform Provides a Simulation Environment Such That the Software Does Not Know That It Is Not Running On Hardware



<https://www.sifive.com>

Imperas U54-MC Virtual Platform




Under 10 seconds to get to booted Linux login prompt!

RISC-V EPK based on Andes N25 (RV32IMAC)

- Extendable Platform Kits (EPKs) are virtual platforms, with software running, to help users start quickly
- EPKs include
 - Individual models, binary and source
 - Platform model, binary and source
 - Software and/or OS running on platform

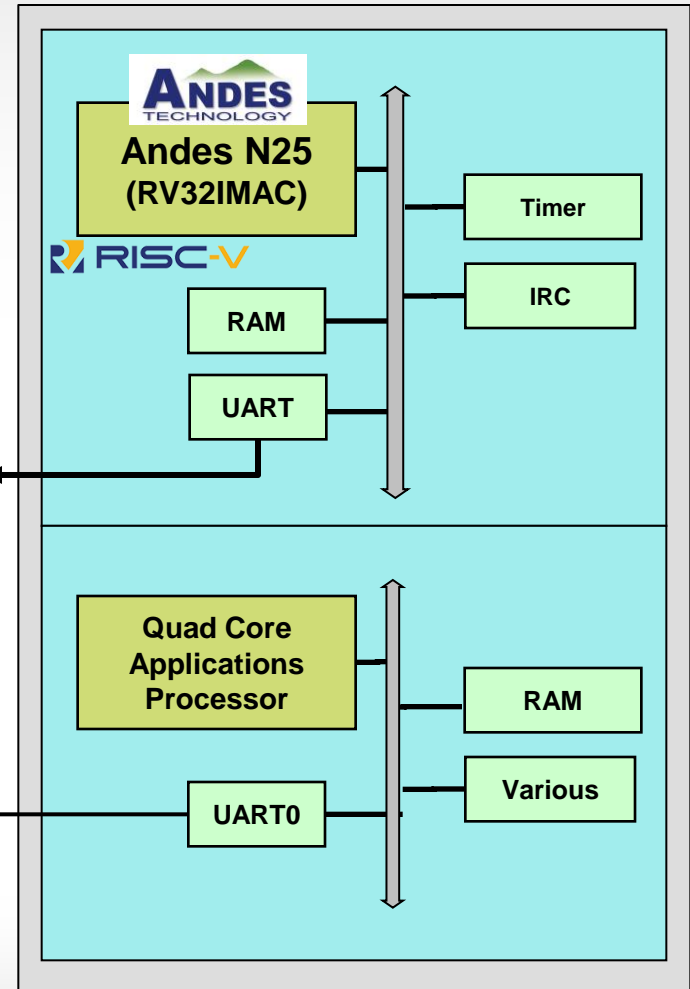
```

ex1@uan0
CPU 7 Task 1 iter= 4
CPU 7 / Task 1 Start Count
CPU 7 / Task 1 End Count
CPU 7 Task 1 iter= 5
CPU 7 / Task 1 Start Count
CPU 7 / Task 2 End Count Anton Test
CPU 7 Task 2 iter= 3
CPU 7 / Task 2 Start Count Anton Test
CPU 7 / Task :
CPU 7 Task 1
CPU 7 / Task :
CPU 7 / Task :
CPU 7 / Task :
CPU 7 Task 2
CPU 7 / Task :
CPU 7 Task 1
CPU 7 / Task :
CPU 7 / Task :
CPU 7 / Task :
CPU 7 Task 1
CPU 7 / Task 1 Start Count
CPU 7 / Task 2 End Count Anton Test
CPU 7 Task 2 iter= 5
CPU 7 / Task 2 Start Count Anton Test
    
```



```

Armv8a64@epk-C4@uan0
v2m_cfg_write: writing 00000002 to 00000000
Console: switching to colour frame buffer device 128x48
smc@f1x: Device version 2000-10-21
smc@f1x: Device not READY in 100ms aborting
isp1760 isp1760: MFP ISP1760 USB Host Controller
isp1760 isp1760: new USB bus registered, assigned bus number 1
isp1760 isp1760: can't setup
isp1760 isp1760: USB bus 1 deregistered
isp1760: failed to register the MFD device
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
USB Mass Storage support registered.
usbcore: registered new interface driver usb-storage
rtc-p1801 rtc: rtc core: registered p1801 as rtc8
smc-p1801 rtc: rtc core: registered p1801 as rtc8
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
oprofile: using arm/oprof7-ca?
TCP: cubic registered
NET: Registered protocol family 17
UFP support v0.11: implementor 41 architecture 3 part 30 variant 7 rev 2
rtc-p1801 rtc: setting system clock to 2011-01-01 00:00:00 UTC (1293840000)
ARM device list:
- No soundcards found.
Freeing init memory: 108K
#####
# Starting benchmarks #
#####
input: AT Raw Set 2 keyboard as /devices/virtual/input/input0
This root FS contains most basic linux utilities (implemented with busybox)
and the linux web browser.
Kernel config is available through /proc/config.gz
Welcome to OUP simulation from Imperas
Log in as root with no password.
Imperas login: input: InExPS/2 Generic Explorer Mouse as /devices/virtual/input/input1
    
```

RISC-V EPK Custom Extensions

- Easy description of Custom Instruction extensions
- No disruption to existing underlying verified model

```

//
// Create the RISC-V decode table
//
static vmidDecodeTableP createDecodeTable(void) {

    vmidDecodeTableP table = vmidNewDecodeTable(RISCV_BITS, RISCV_EIT_LAST);

    // opcode [6:0] = 00 010 11
    // func3[14:12] = 1,2,3,4 (QR1-4)
    // rs1[19:15]
    // rs2[24:20]
    // dst[11:7]

    // handle exchange instruction
    DECODE_ENTRY(0, CHACHA20QR1, ".000.....0001011");
    DECODE_ENTRY(0, CHACHA20QR2, ".001.....0001011");
    DECODE_ENTRY(0, CHACHA20QR3, ".010.....0001011");
    DECODE_ENTRY(0, CHACHA20QR4, ".011.....0001011");

    return table;
}

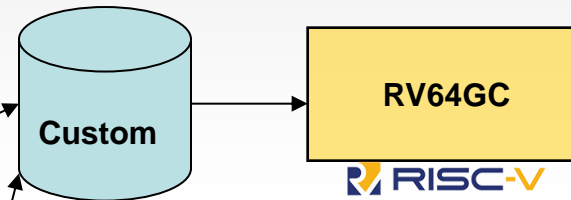
//
// Emit code implementing exchange instruction
//
static void emitChaCha20(
    vmiProcessorP processor,
    vmiosObjectP object,
    Uns32 instruction,
    Uns32 rot1
) {

    // extract instruction fields
    Uns32 rd = RD(instruction);
    Uns32 rs1 = RS1(instruction);
    Uns32 rs2 = RS2(instruction);

    vmiReg reg_rs1 = vmimtGetExtReg(processor, &object->rs1);
    vmiReg reg_rs2 = vmimtGetExtReg(processor, &object->rs2);
    vmiReg reg_tmp = vmimtGetExtTemp(processor, &object->tmp);

    vmimtGetR(processor, 64, reg_rs1, object->riscvRegs[rs1]);
    vmimtGetR(processor, 64, reg_rs2, object->riscvRegs[rs2]);
    vmimtBinopRRR(32, vmi_XOR, reg_tmp, reg_rs1, reg_rs2, 0);
    vmimtBinopRC(32, vmi_ROL, reg_tmp, rot1, 0);

    vmimtSetR(processor, 64, object->riscvRegs[rd], reg_tmp);
}
    
```



- ChaCha20 cipher as example of custom instructions for algorithm accelerators
- Instruction Extensions to RISC-V courtesy of Cerberus Security Laboratories Ltd
- <https://cerberus-laboratories.com>

RISC-V EPK Missing Custom Extensions (FU540)

```

root@ucbvax:~# ./custom.sh
Running C Algorithm Implementation
RES = 84772366
real 0m 3.32s
user 0m 3.23s
sys 0m 0.08s

Running ASM Algorithm Implementation
[ 12.945696] custom_asm.exe[80]: unhandled signal 4 code 0x1 at 0x00000000010402 in custom_asm.exe[1000+6b000]
[ 12.945696] CPU: 4 PID: 80 Comm: custom_asm.exe Tainted: G W 4.15.0-00044-g2b0aa1d #2
[ 12.945800] sepc: 000000000010402 ra : 00000000001043a sp : 000003ffff9b9c40
[ 12.945800] gp : 000000000007e0f0 tp : 0000000000080700 t0 : 0000000000000000
[ 12.945800] t1 : 00000000000210de t2 : 0000000000001000 s0 : 000003ffff9b9c70
[ 12.945800] s1 : 0000000000010a36 a0 : ffffffff84772366 a1 : 000000004c1fba97
[ 12.945900] a2 : 0000000000000004 a3 : 0000000000000001 a4 : 0000000000000001
[ 12.945900] a5 : 000000004c1fba97 a6 : 0000000000000000 a7 : 000003ffff9b9c4c
[ 12.945900] s2 : 0000000000000000 s3 : 000003ffff9c2dd0 s4 : 0000000000000002
[ 12.946000] s5 : 0000003ffff9c2f55 s6 : 00000000000f0118 s7 : 0000000000000002
[ 12.946000] s8 : 0000000000000014 s9 : 0000020001b8720 s10: 0000000000000000
[ 12.946000] s11: 0000003ffff41fb6 t3 : 000000000000006d t4 : 0000000000000000
[ 12.946000] t5 : 0000000000000062 t6 : 0000000000000078
[ 12.946100] sstatus: 000000200002020 sbadaddr: 000000000b5050b scause: 000000000000002
Command terminated by signal 4
real 0m 0.02s
user 0m 0.00s
sys 0m 0.02s
    
```

Linux
Console

Virtual Platform
Console

```

Info 'FU540/U54_hart1', 0x0000000000015a76(fread+9e): 8082 jr ra
Info 'FU540/U54_hart1', 0x000000000001043a(main+80): 87aa mv a5,a0
Info 'FU540/U54_hart1', 0x000000000001043c(main+82): ffcdbnez a5,103f6
Info 'FU540/U54_hart1', 0x00000000000103f6(main+3c): fec42783 lw a5,-20(s0)
Info 'FU540/U54_hart1', 0x00000000000103fa(main+40): 853e mv a0,a5
Info 'FU540/U54_hart1', 0x00000000000103fc(main+42): fd842783 lw a5,-40(s0)
Info 'FU540/U54_hart1', 0x0000000000010400(main+46): 85be mv a1,a5
Info 'FU540/U54_hart1', 0x0000000000010402(main+48): 00b5050b undef
Info 'FU540/U54_hart1', 0x0000000080000004(trap_vector): 34011173 csrwr sp,mscratch,sp
Info 'FU540/U54_hart1', 0x0000000080000008(trap_vector+4): 1a010863 beqz sp,800001b8
Info 'FU540/U54_hart1', 0x000000008000000c(trap_vector+8): 04a13823 sd a0,80(sp)
Info 'FU540/U54_hart1', 0x0000000080000010(trap_vector+c): 04b13c23 sd a1,88(sp)
Info 'FU540/U54_hart1', 0x0000000080000014(trap_vector+10): 342025f3 csr a1,mcause
Info 'FU540/U54_hart1', 0x0000000080000018(trap_vector+14): 0805d263 bgez a1,8000009c
Info 'FU540/U54_hart1', 0x000000008000009c(trap_vector+98): 00113423 sd ra,8(sp)
Info 'FU540/U54_hart1', 0x00000000800000a0(trap_vector+9c): 00313c23 sd gp,24(sp)
Info 'FU540/U54_hart1', 0x00000000800000a4(trap_vector+a0): 02413023 sd tp,32(sp)
    
```

RISC-V EPK Implemented Custom Extensions (FU540)

```

ucbvax login: root
Password:

root@ucbvax:~# ./custom.sh
Running C Algorithm Implementation
RES = 84772366
real 0m 3.32s
user 0m 3.26s
sys 0m 0.05s

Running ASM Algorithm Implementation
RES = 84772366
real 0m 1.53s
user 0m 1.45s
sys 0m 0.08s
root@ucbvax:~#
    
```

Linux Console

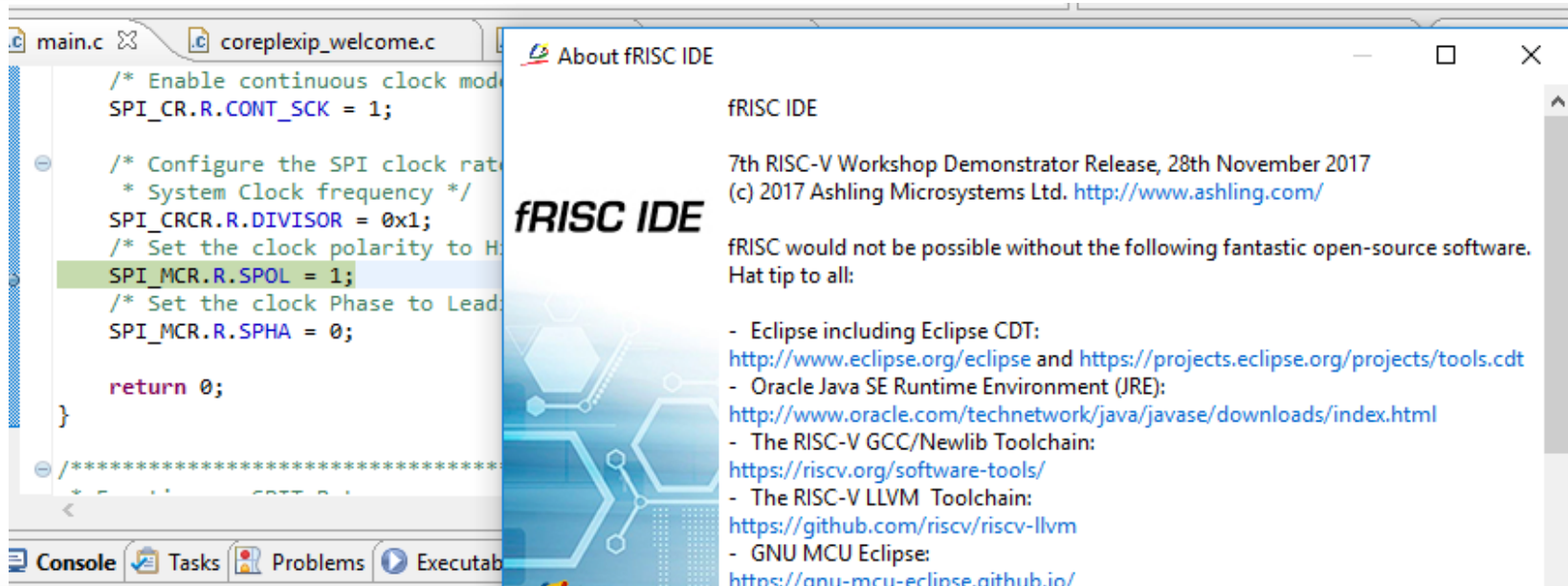
```

Info 'FU540/U54_hart1', 0x0000000000015a76(fread+9e): 8082    jr    ra
Info 'FU540/U54_hart1', 0x000000000001043a(main+80): 87aa    mv    a5,a0
Info 'FU540/U54_hart1', 0x000000000001043c(main+82): ffc0    bnez  a5,103f6
Info 'FU540/U54_hart1', 0x00000000000103f6(main+3c): fec42783 lw    a5,-20(s0)
Info 'FU540/U54_hart1', 0x00000000000103fa(main+40): 853e    mv    a0,a5
Info 'FU540/U54_hart1', 0x00000000000103fc(main+42): fd842783 lw    a5,-40(s0)
Info 'FU540/U54_hart1', 0x0000000000010400(main+46): 85be    mv    a1,a5
Info 'FU540/U54_hart1', 0x0000000000010402(main+48):         chacha20qr1 a0,a0,a1
Info 'FU540/U54_hart1', 0x0000000000010406(main+4c):         chacha20qr2 a0,a0,a1
Info 'FU540/U54_hart1', 0x000000000001040a(main+50):         chacha20qr3 a0,a0,a1
Info 'FU540/U54_hart1', 0x000000000001040e(main+54):         chacha20qr4 a0,a0,a1
Info 'FU540/U54_hart1', 0x0000000000010412(main+58):         chacha20qr1 a0,a0,a1
Info 'FU540/U54_hart1', 0x0000000000010416(main+5c):         chacha20qr2 a0,a0,a1
Info 'FU540/U54_hart1', 0x000000000001041a(main+60):         chacha20qr3 a0,a0,a1
Info 'FU540/U54_hart1', 0x000000000001041e(main+64):         chacha20qr4 a0,a0,a1
Info 'FU540/U54_hart1', 0x0000000000010422(main+68): 87aa    mv    a5,a0
Info 'FU540/U54_hart1', 0x0000000000010424(main+6a): fef42623 sw    a5,-20(s0)
Info 'FU540/U54_hart1', 0x0000000000010428(main+6e): fd840793 addi  a5,s0,-40
Info 'FU540/U54_hart1', 0x000000000001042c(main+72): fe043683 ld    a3,-32(s0)
Info 'FU540/U54_hart1', 0x0000000000010430(main+76): 4605    li    a2,1
Info 'FU540/U54_hart1', 0x0000000000010432(main+78): 4591    li    a1,4
    
```

Virtual Platform Console

Software Development using Ashling RISC-V IDE

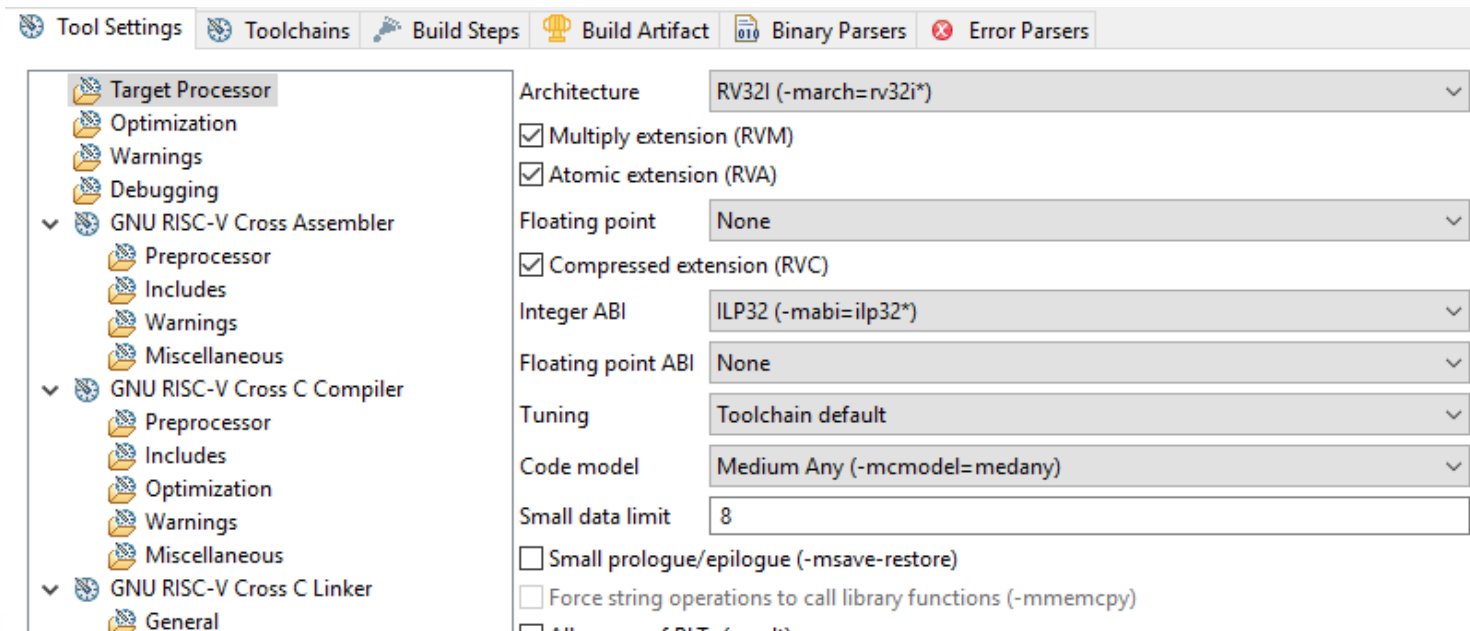
- Ashling RISC-V IDE Integrates with Imperas Virtual Platforms/Processor Models
- IDE supports full software development cycle including edit, build, debug, test and verification on the actual Virtual Platform or Processor Model all from a user-friendly Eclipse based IDE environment



Software Development using Ashling RISC-V IDE

cont'd

- The same software toolchain/IDE can be used throughout the complete design cycle....from simulation using the Imperas models to device/board bring-up with actual silicon
- Includes latest RISC-V compilers including GCC and LLVM



The screenshot shows the 'Tool Settings' window in the Ashling RISC-V IDE. The 'Toolchains' tab is selected. The left sidebar shows a tree view of toolchain settings, with 'GNU RISC-V Cross Assembler' and 'GNU RISC-V Cross C Compiler' expanded. The main panel displays configuration options for the selected toolchain:

- Architecture: RV32I (-march=rv32i*)
- Multiply extension (RVM)
- Atomic extension (RVA)
- Floating point: None
- Compressed extension (RVC)
- Integer ABI: ILP32 (-mabi=ilp32*)
- Floating point ABI: None
- Tuning: Toolchain default
- Code model: Medium Any (-mcmmodel=medany)
- Small data limit: 8
- Small prologue/epilogue (-msave-restore)
- Force string operations to call library functions (-mmemcpy)

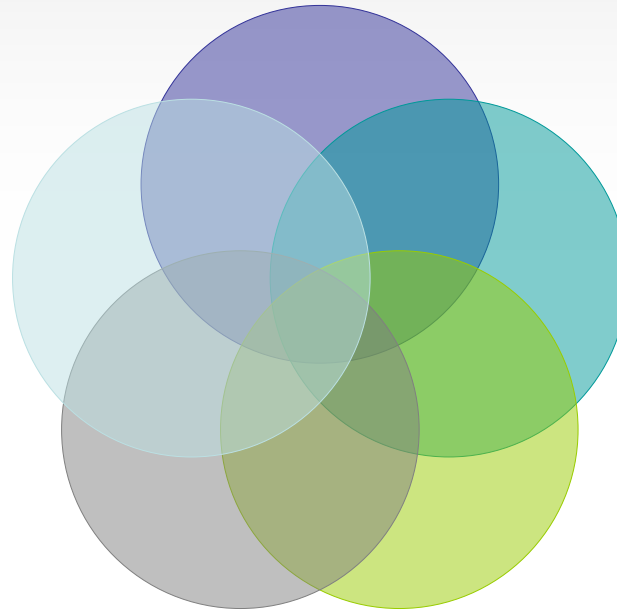
Imperas & Ashling solutions

Methodology

Collaboration with customers, vendor ecosystem

Models

200+ CPU models
 200+ peripheral models
 50+ EPK (Extendable Platform Kits)



Tools

Leading simulation, debug, software verification tools

Resources

Imperas and partners
 Model development
 Tool development

Training

Imperas and partners
 On-site, customized agenda

Virtual Platforms Accelerate Software Development

- Risk-free addition of custom instruction extensions without disrupting model quality
- Complete the virtual prototype before silicon or even RTL is available
- Accelerate software development and porting of existing software
- Use EPK for
 - Ecosystem partners
 - Early application development
 - Lead customers

Contact

Hugh O'Keeffe

Engineering Director

hugh.okeeffe@ashling.com

Lee Moore

Applications Engineering

moore@imperas.com

Kevin McDermott

kevinm@imperas.com

<http://www.ashling.com>

<http://www.imperas.com>

<http://www.ovpworld.org>