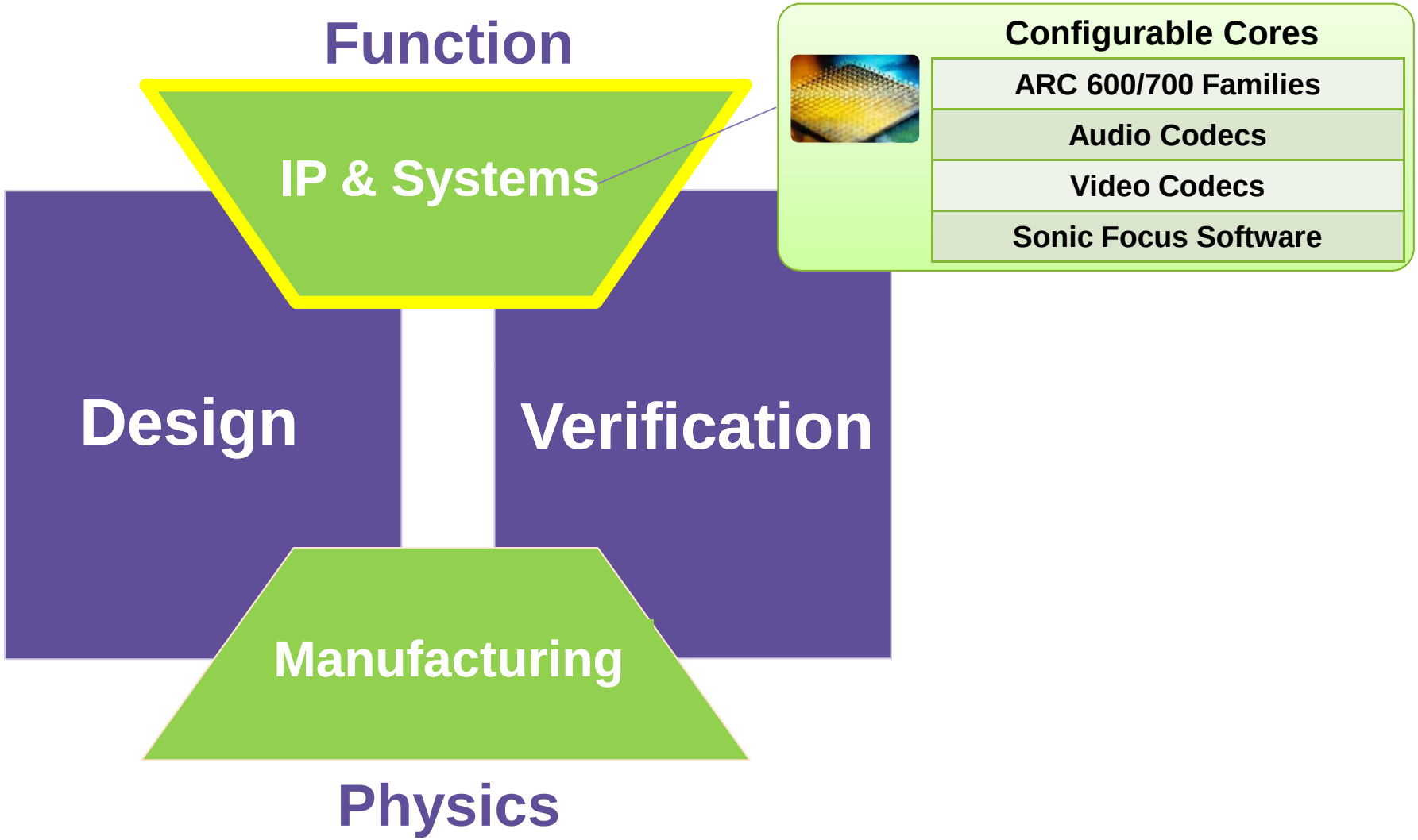
Experience with GNU, LINUX, and other Open Source on ARC Processors

Portability Is For People Who Cannot Write New Programs

ELC-Europe 2010, Cambridge, October 27/28

Mischa Jonker, Ruud Derwig

Synopsys & ARC Introduction



DesignWare ARC Processors

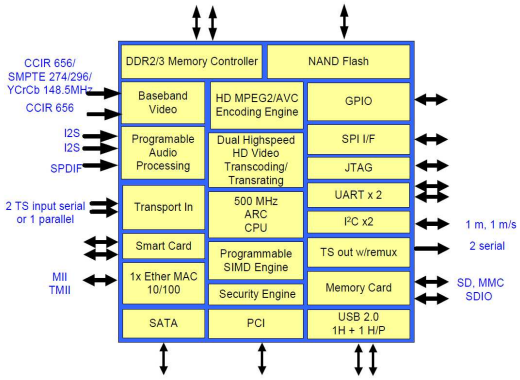
A History of Customer Success

- #2 supplier of embedded 32-bit Processor IP cores
  - 160+ licensees (7 of the top 10 semis)
  - >550 million ARC-based™ chips shipping annually
  - Thousands of successful customer tapeouts
- Complete Multimedia Solutions
- Configurable & Extendible Technologies to meet your needs
- Full suite of software & hardware development tools
- The Most Size and Power Efficient Cores!
  - 1.52 DMIPS/MHz, 0.13mw/MHz, 11.7 DMIPS/mW

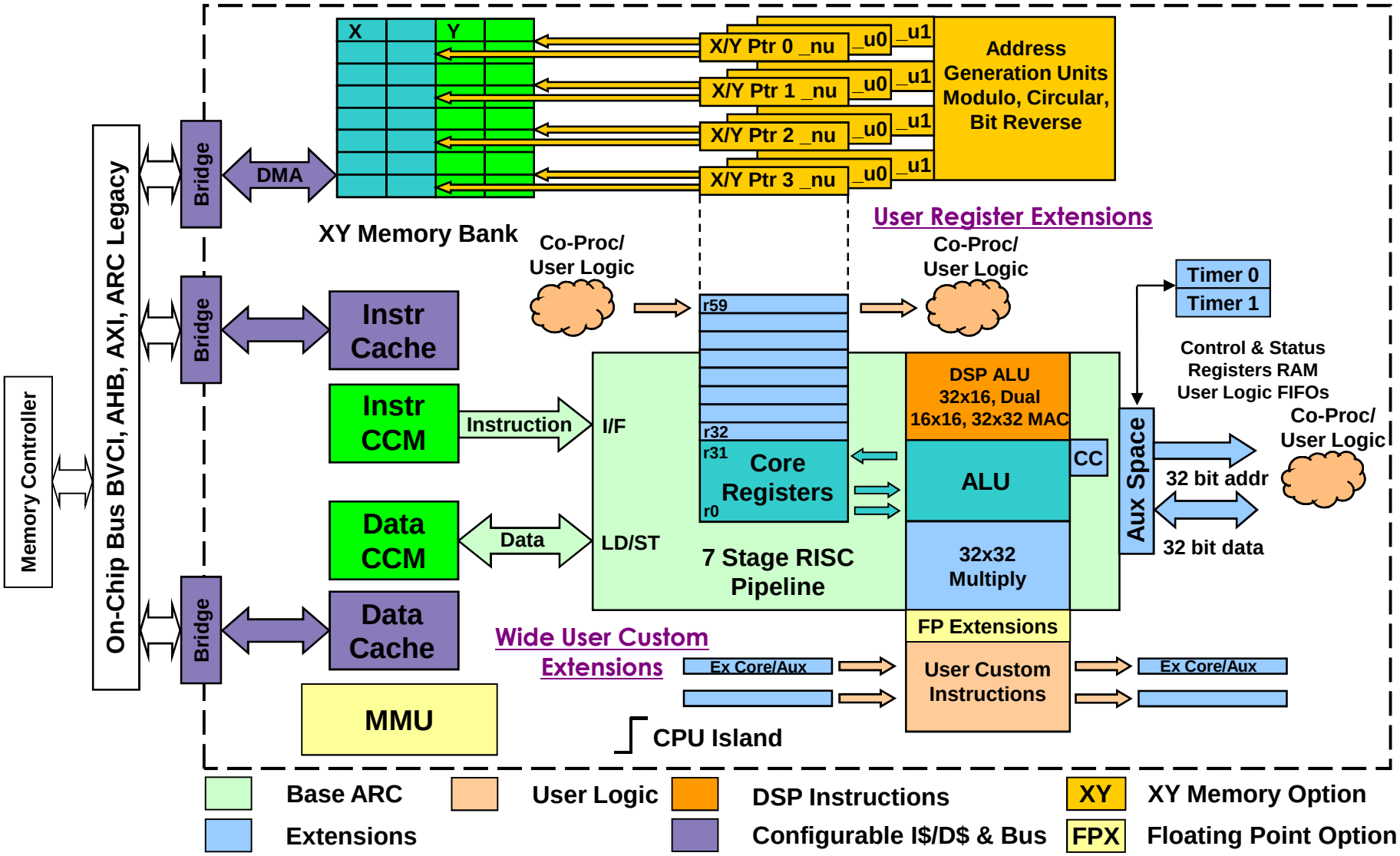
Some example products …

- **2.5" 1080p FULLHD Player, MP4 Player**
- **3.5" SATA 1080p FULL HD Network Media Player**
  (based on AMLogic chipset)

- **DVR, STB players**
  (based on ViXS chipset)



DW ARC 7xx – Configurable & Extendable



DW ARC 7xx – Configurable & Extendable

- Targeting host and application processors: up to 1.2GHz at 40G delivering 1800 DMIPS
- 7 stage scalar, interlocked pipeline, with Dynamic Branch Prediction

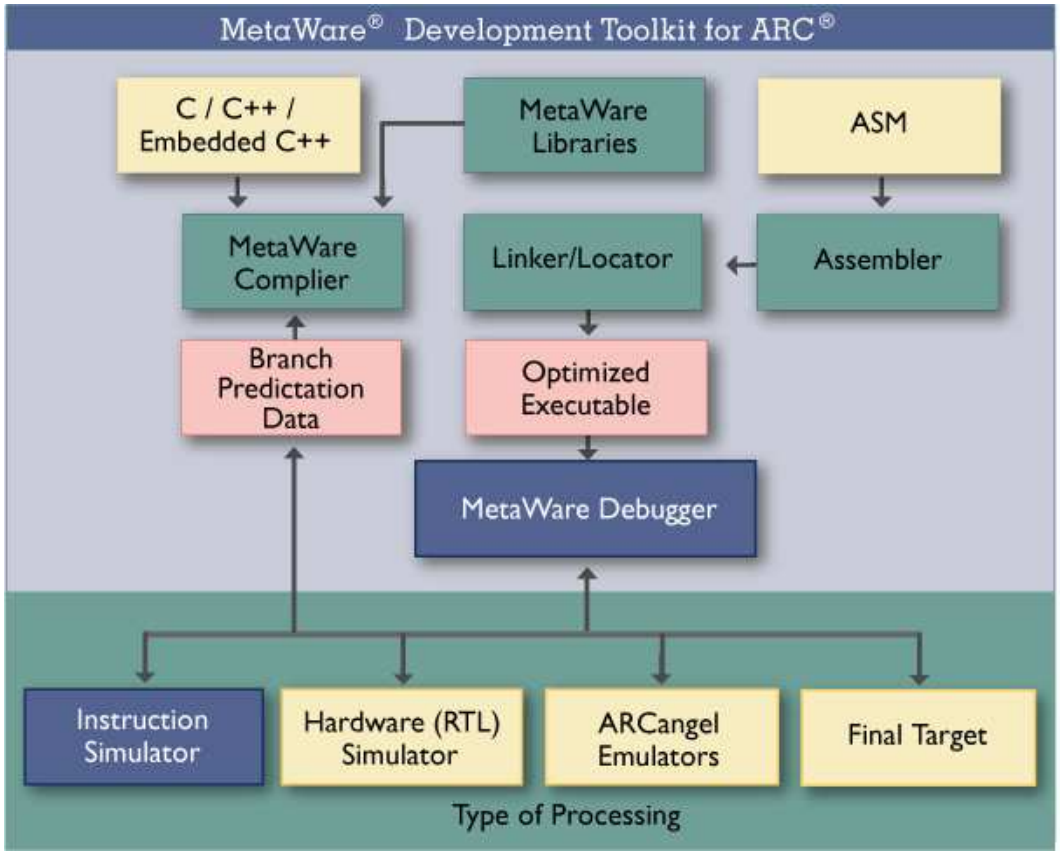
  (with 512 entry branch target address cache)
- 64KB I-cache/D-cache
- MMU with 128 entry 2-way set associative main TLB
- Configurable bus masters (BVCI/AHB/AXI, 32/64 bit)
- “Direct Memory Interface” for instruction and data CCMs
- ARCompact ISA : hybrid 16/32 bit instructions for minimal code size
- DSP extensions and customs instructions: customers can define their own application specific instructions to accelerate

Overview of porting steps

- Toolchain
- C runtime library
- Kernel
- Userland packages
- Distribution / delivery
- The return path

Toolchain

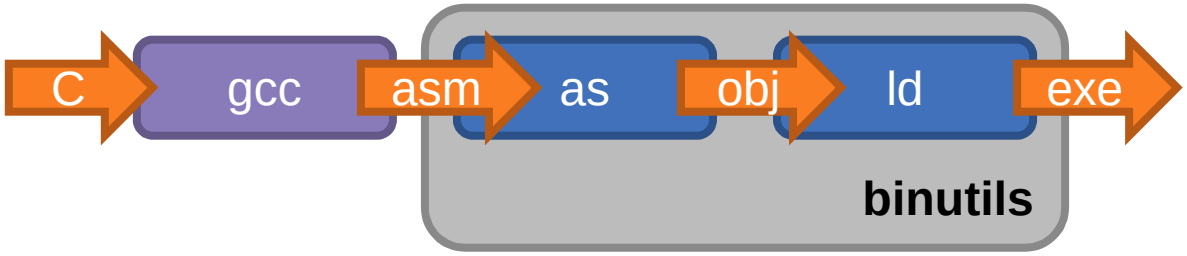
- Metaware toolchain
  - Proven technology;
  - Adapted to ARC architecture: makes use of most ARC-specific enhancements;
  - Not very suitable for building Linux, because of usage of GCC extensions:

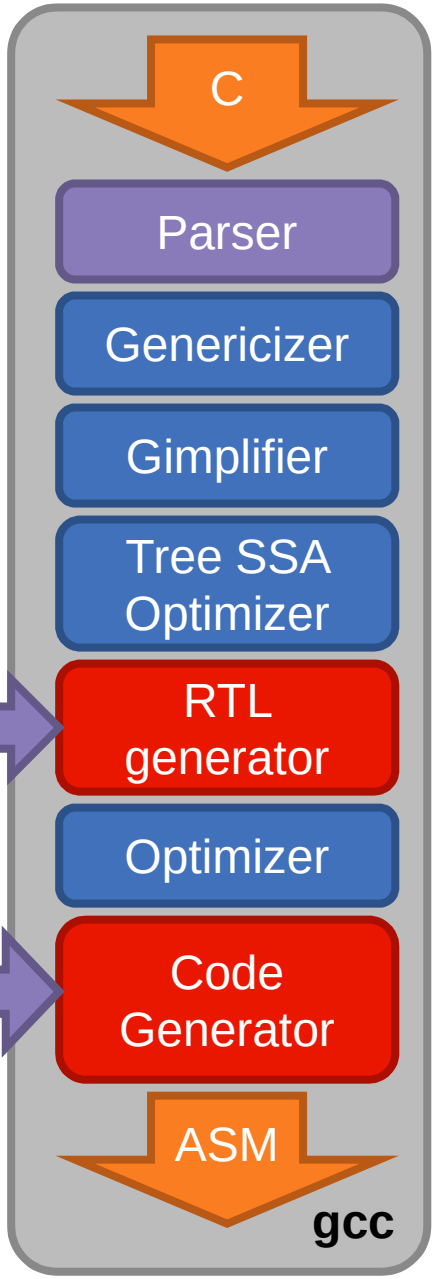



```
switch (major_idx) {
case 0:         return SCSI_DISK0_MAJOR;
case 1 ... 7:   return SCSI_DISK1_MAJOR + major_idx - 1;
case 8 ... 15:  return SCSI_DISK8_MAJOR + major_idx - 8;
default:        BUG();
                return 0;
}
```

Toolchain: porting GCC

- Machine description:
  - **gcc/config/arc/arc.md**: contains templates that convert from an intermediate representation to actual ARC assembly.
  - **gcc/config/arc/arc.c**: the description often uses C functions to assist in generating ASM.
  - **gcc/config/arc/ieee-754**: handlers for software floating point
- Optimization/tuning:
  - **MilePost**: machine learning and iterative feedback mechanism for finding the optimum tuning options for a specific architecture
- Key issues:
  - Correctness of code, performance





Example

- Brcc instruction: compare and branch:
  - Limited range, when out of range needs to be split into cmp and bcc

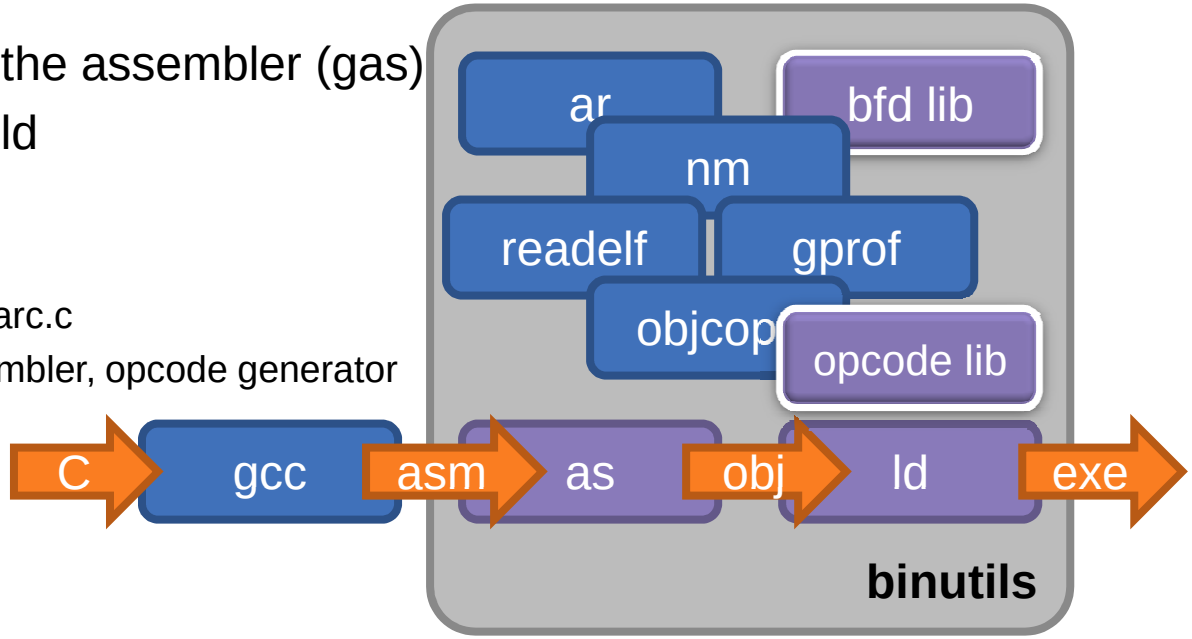
```
breq.d r1, 0, .Label
sub.ne r2, r2, 1
```

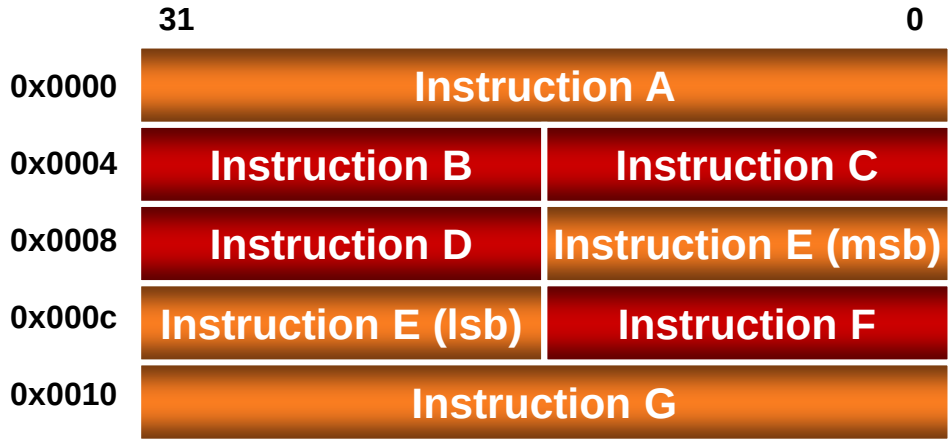
```
cmp    r1, 0
beq.d  r1, 0, .Label
sub.ne r2, r2, 1
```

Conditional instruction fails because of extra cmp

Toolchain: porting binutils

- Adding support for ARC in the assembler (gas)
- Adding support for ARC in ld
  - Adding linker scripts
- But also:
  - BFD library: cpu-arc.c, elf32-arc.c
  - Opcode library: ARC disassembler, opcode generator
- 16/32 bits instructions:
  - Mixed endianness;
  - Unaligned relocations

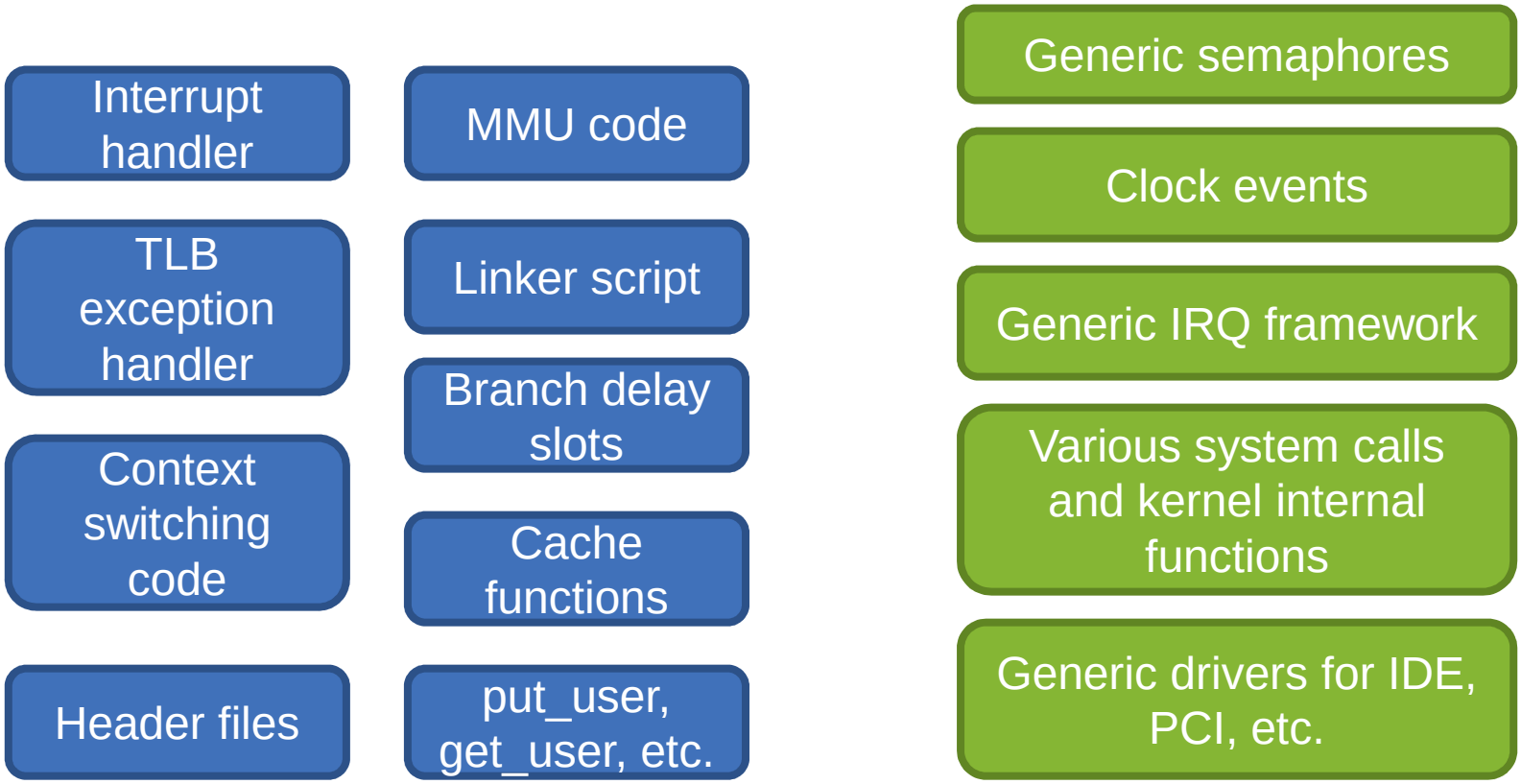




Runtime libraries: uClibc

- Just compile and go… not:-(
- C-runtime entry code
- Architecture-specific header files
- Dynamic linker, difference between code/data
  - Relocatable code, little/big endian 16/32 bit
- System call handlers
- Hand-optimized memcpy, strcpy, etc.
- Hand-written implementations for longjmp, but also some system calls like clone()
- Handlers for atomic operations
- TLS

The kernel

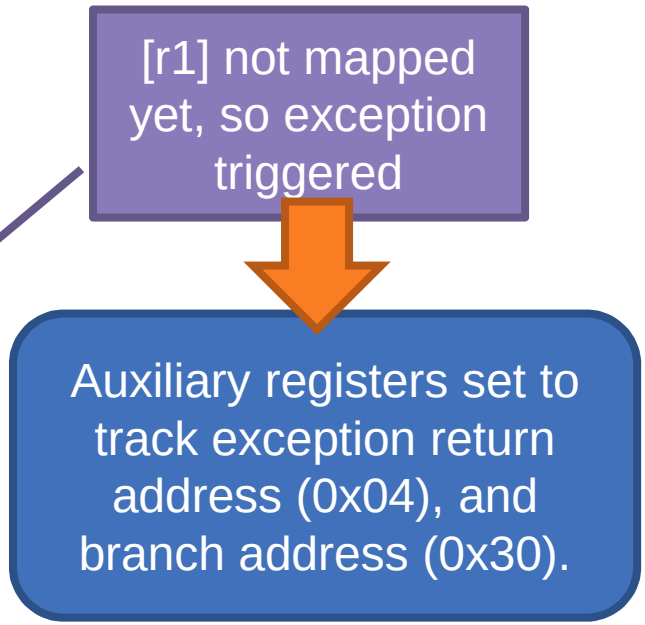


Architecture specific changes

Reuse of generic kernel components

Example (branch delay slot):

```
0x00: b.d     0x30
0x04: ld      r0, [r1]
...
0x30:
```



What happens if we get a task switch now??

Make sure that complete processor status is saved and restored…

Userland

- Alignment issues, hidden assumptions (i.e. 8KB page size)
- Thread local storage → emulation
- Linking sometimes needs tuning with large objects
- Architecture-specific code in userland packages
- Example: Webkit: knows about ARM, but not about ARC

```
    // FIXME: perhaps we should have a more abstract macro that indicates when
    // going 4 bytes at a time is unsafe
#if CPU(ARM) || CPU(SH4)
    const UChar* stringCharacters = string->characters();
    for (unsigned i = 0; i != length; ++i) {
        if (*stringCharacters++ != *characters++)
            return false;
    }
    return true;
#else
    /* Do it 4-bytes-at-a-time on architectures where it's safe */

    const uint32_t* stringCharacters = reinterpret_cast<const uint32_t*>(string->characters());
    const uint32_t* bufferCharacters = reinterpret_cast<const uint32_t*>(characters);
```

Distribution / delivery

- Customers expect more than kernel + toolchain
- Many (all) customers have their own preference
  - What flavor to choose?
- There's a lot to leverage from OSS community
- Started with minimal approach:
  - Kernel, toolchain and minimal Root FS on SourceForge;
  - Supporting customers in porting their libraries.
- Will be extended over time
  - Extend distribution by adding packages and libraries

The return path

- Being a good citizen… or sound business sense?
  - ARC GCC 2.3 and ARC Linux 1.3 (based on Linux 2.6.30 and GCC 4.2.1) available at: http://sourceforge.net/projects/arc-linux/
- Next steps:
  - Get ARC Linux merged into Linux kernel
  - Learn to play according the OSS community rules

Future work

- Linux audio/video solutions, including highly optimized codecs
  - ARC can reduce porting effort because of the single processor architecture for both host (ARC Linux) and DSP (ARC Media subsystems).
- Mainlining Linux kernel and tools

Conclusions

- ARC and Linux work together and form a nice alternative with size, power and configurability benefits
- In general: the further down the chain you get, the easier it gets to port it;
- However, core-specific optimizations and assumptions are often well hidden and require some debugging/porting effort;

- Portability is for people who cannot write new programs; verdict: busted/plausible/confirmed

SYNOPSYS®

Predictable Success