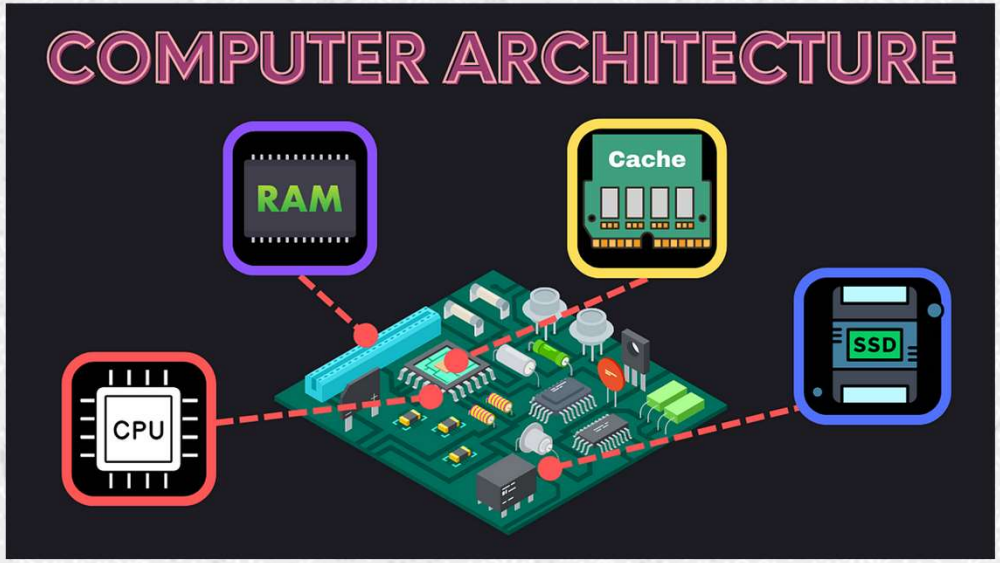


CLOCKHANDS: RENAME-FREE INSTRUCTION SET ARCHITECTURE FOR OUT-OF-ORDER PROCESSORS

Promod Chandra Das
Id: 231002005

Seam Mrida
Id:231002016

❖ ABSTRACT

The paper "Clockhands: Rename-free Instruction Set Architecture for Out-of-order Processors" presents a novel approach to eliminate register renaming in out-of-order processors. It introduces a mechanism called "Clockhands," which allows precise allocation and deallocation of physical registers without traditional renaming techniques. By simplifying the process of mapping logical registers to physical ones, Clockhands reduces complexity and improves processor performance. The technique also enhances scalability by avoiding common bottlenecks associated with register renaming, particularly in high-performance architectures. Experimental results demonstrate that the Clockhands method achieves comparable performance to conventional designs while simplifying hardware implementation.

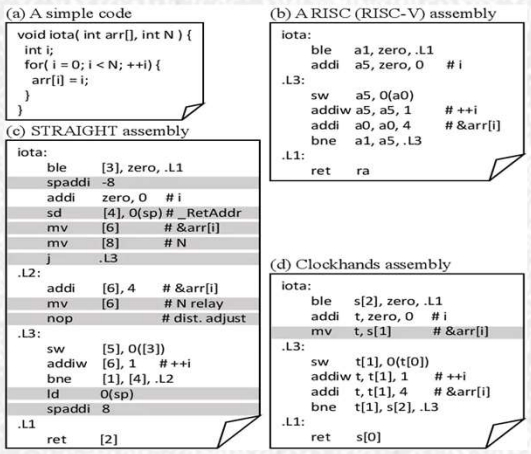
CCS CONCEPTS: Computer systems organization Superscalar architecture; *Reduced instruction set computing*; • Software and its engineering → *Compilers*.

KEYWORDS

Instruction : set architecture, Superscalar processor, Out-of-order execution, Register renaming, Compiler, Power efficiency, Register lifetime

INTRODUCTION

The introduction of "Clockhands: Rename-free Instruction Set Architecture for Out-of-order Processors" highlights the limitations of traditional register renaming in modern processors, particularly its complexity and performance bottlenecks. The paper proposes the Clockhands mechanism as a simpler, efficient alternative, aiming to streamline register management while maintaining high performance in out-of-order execution.



(a) A simple code

```
void iota( int arr[], int N ) {
  int i;
  for( i = 0; i < N; ++i) {
    arr[i] = i;
  }
}
```

(b) A RISC (RISC-V) assembly

```
iota:
      ble     a1, zero, .L1
      addi    a5, zero, 0     # i
.L3:
      sw      a5, 0(a0)
      addiw a5, a5, 1         # ++i
      addi    a0, a0, 4       # &arr[i]
      bne     a1, a5, .L3
.L1:
      ret     ra
```

(c) STRAIGHT assembly

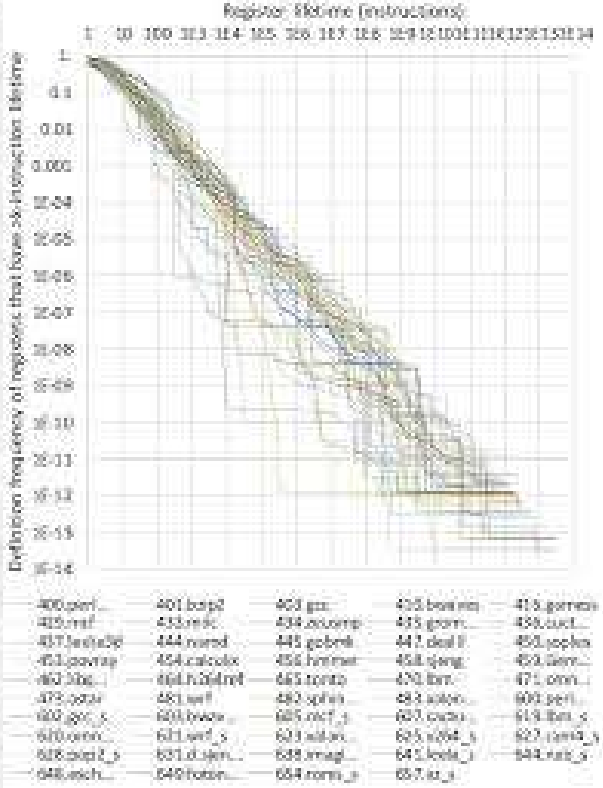
```
iota:
      ble     [3], zero, .L1
      spaddi  -8
      addi    zero, 0     # i
      sd      [4], 0(sp) # _RetAddr
      mv      [6]         # &arr[i]
      mv      [8]         # N
      j       .L3
.L2:
      addi    [6], 4      # &arr[i]
      mv      [6]         # N relay
      nop                 # dist. adjust
.L3:
      sw      [5], 0([3])
      addiw   [6], 1      # ++i
      bne     [1], [4], .L2
      ld      0(sp)
      spaddi  8
.L1
      ret     [2]
```

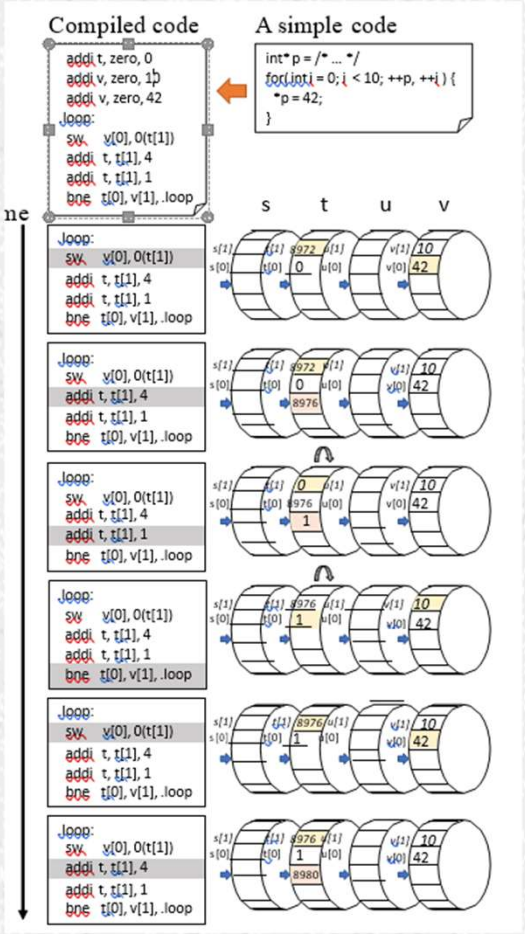
(d) Clockhands assembly

```
iota:
      ble     s[2], zero, .L1
      addi    t, zero, 0    # i
      mv      t, s[1]       # &arr[i]
.L3:
      sw      t[1], 0(t[0])
      addiw t, t[1], 1      # ++i
      addi    t, t[1], 4    # &arr[i]
      bne     t[1], s[2], .L3
.L1:
      ret     s[0]
```

EXISTING OPERAND SPECIFICATION

The "Existing Operand Specification" section discusses how current out-of-order processors use register renaming to handle operand mapping, which introduces complexity and resource overhead. Traditional designs rely on renaming tables to track physical registers, leading to performance limitations. This approach often complicates operand fetching, instruction scheduling, and increases power consumption.





CLOCKHANDS OVERVIEW

The "Clockhands Overview" section introduces the Clockhands mechanism, which eliminates the need for traditional register renaming in out-of-order processors. Clockhands manages register allocation and deallocation using a circular queue-like structure. This approach simplifies physical register tracking, reducing complexity and overhead while maintaining high performance by enabling efficient operand handling and instruction execution.

RENAME-FREE INSTRUCTION SET ARCHITECTURE FOR OUT-OF-ORDER PROCESSORS

CLOCKHANDS ISA

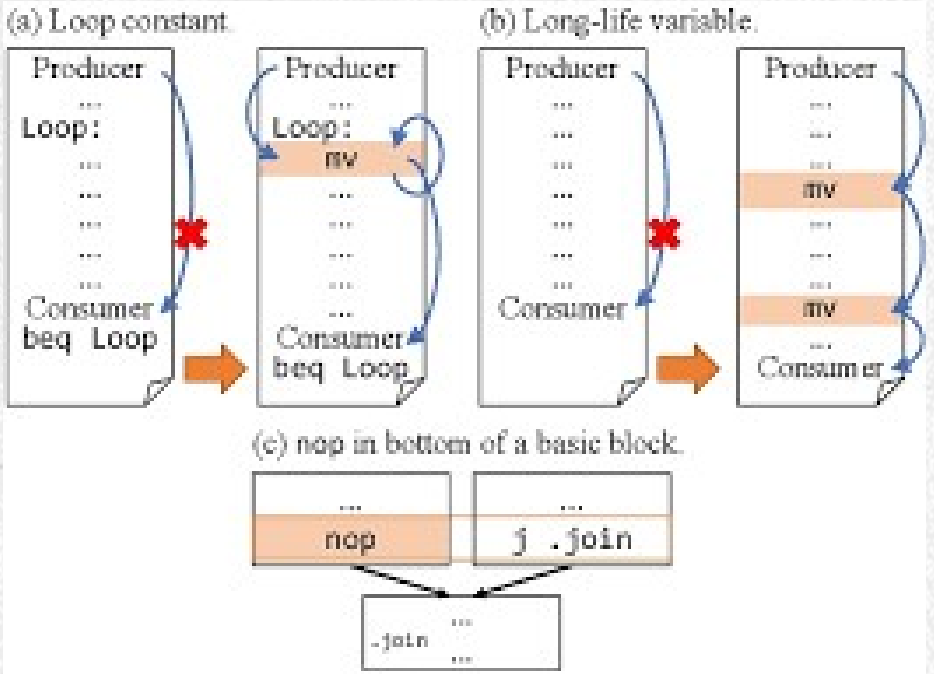
The "Clockhands ISA" section describes the instruction set architecture modifications required for the Clockhands mechanism. It adapts existing ISA structures to support efficient physical register management without traditional renaming, enabling streamlined execution.

MICROARCHITECTURE

The "Microarchitecture" section details how the Clockhands mechanism integrates into processor design. It outlines key components, such as the circular register queue, operand handling units, and their interaction for efficient execution.

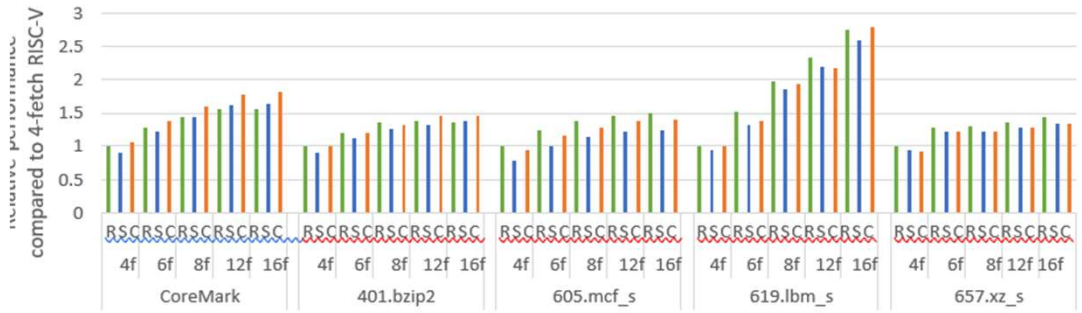
CLOCKHANDS COMPILER

The "Clockhands Compiler" section explains how the compiler is adapted to work with the Clockhands mechanism. It optimizes instruction scheduling and register allocation, ensuring efficient mapping of logical registers to physical registers without renaming.



EVALUATION

The "Evaluation" section presents the performance analysis of the Clockhands mechanism. Simulations compare Clockhands to conventional out-of-order processors with register renaming, demonstrating similar or improved performance. The results highlight reduced hardware complexity, lower power consumption, and scalability benefits. Clockhands proves to be effective in maintaining high throughput while simplifying register management and processor design.



RELATED WORK

The "Related Work" section reviews prior research on register renaming, out-of-order execution, and alternative register management techniques. It contrasts these methods with the Clockhands approach, highlighting its novelty in simplifying processor design.

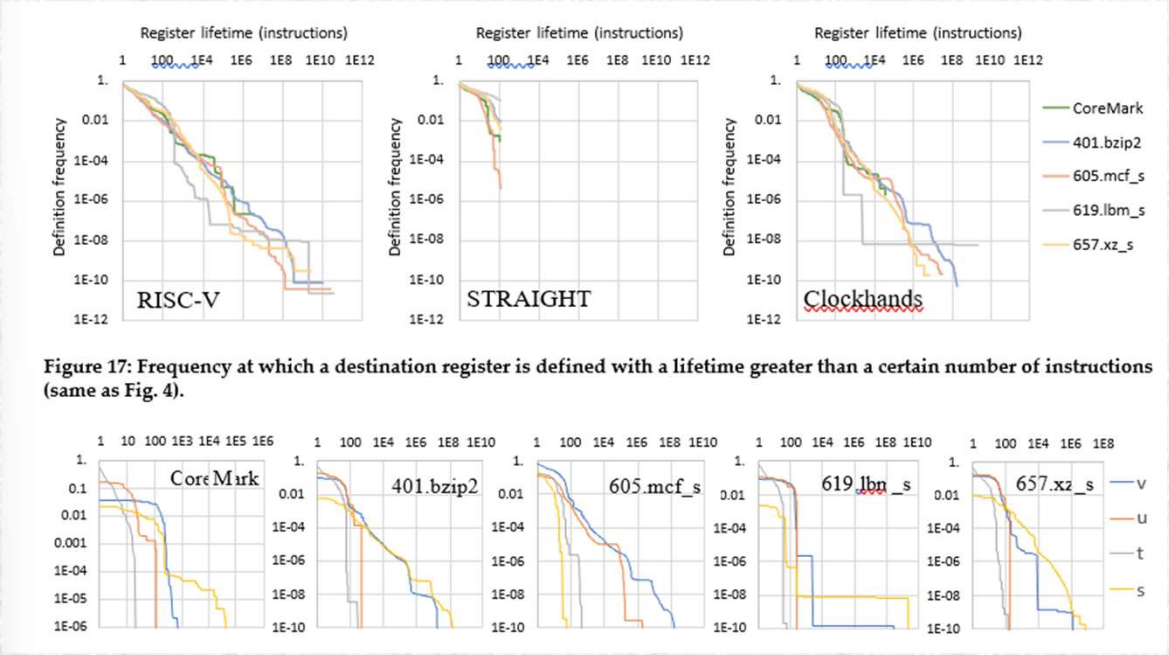


Figure 17: Frequency at which a destination register is defined with a lifetime greater than a certain number of instructions (same as Fig. 4).

CONCLUSION

The "Conclusion" summarizes the benefits of the Clockhands mechanism as a rename-free approach to managing registers in out-of-order processors. By eliminating register renaming, Clockhands simplifies processor design, reduces complexity, and maintains high performance. The method is shown to be a scalable and energy-efficient alternative to traditional register management techniques.

THANK YOU

Have you any question?