Simple Explanation of the Slides

These slides explain how a stack-based CPU works and why memory references happen during stack operations.

1. Stack Machines Do Math Using a Stack

A stack machine doesn't use many registers like R1, R2, R3. Instead, it does operations like:

- push a → put 'a' onto the stack

- push b → put 'b' onto the stack

- * → multiply the top two values on the stack

The expression shown:

a b c * + a d c * + e - /

is in postfix (stack■friendly) notation.

2. Only the Top 2 Items Are Kept in Fast Registers

The CPU keeps:

- Top of stack → R0

- Second item → R1

Everything below R1 goes to memory, which is slower.

3. Memory References Happen Only When Necessary

If too many values are pushed:

- Older values spill to memory → memory reference

When needed again:

- They must be loaded → another memory reference

This is slow, which is why the slide says "No Good!"

4. Goal: Keep as Much of the Stack in Registers as Possible

Performance improves when:

- Top N stack values stay in registers

- Memory is used only on overflow

5. What the Table Shows

The table shows, for each instruction:

- What the stack contains

- Which values are in registers (R0, R1)

- When values spill to memory

Example:

push c → stack becomes R0 R1 R2

Since only R0 and R1 are registers, R2 is stored in memory.

Bottom Line

A stack machine is fastest when:

- The top few stack items fit in registers

- Memory is touched only when necessary

More stack depth → more memory use → slower performance.