IF_ID Stage Module Explained

Purpose

The `IF_IDstage` module implements the pipeline register between the **Instruction Fetch (IF)** and **Instruction Decode (ID)** stages in your 4-stage RISC-V pipeline CPU. It captures and holds the instruction and the program counter (PC) value output from the IF stage at each clock cycle, making both available to the ID stage in the following cycle.

How the Module Work

Signal	Direction	Width	Description
clk	input	1 bit	Clock signal, triggers register update
rst	input	1 bit	Reset signal, clears the register on high
pc_in	input	32 bits	PC value from IF stage
instr_in	input	32 bits	Fetched instruction from IF stage
pc_out	output	32 bits	Latched PC value for ID stage
instr_out	output	32 bits	Latched instruction for ID stage

Behavior

- **On clock rising edge**: The module stores the input PC and instruction into internal outputs (`pc_out` and `instr_out`). These outputs drive the next pipeline stage.
- **On reset (`rst` high)**: The outputs are cleared;
    - `pc_out` is set to zero.
    - `instr_out` is set to `32'h0000_0013`, which is the 32-bit encoding of the RISC-V NOP instruction (`ADDI x0, x0, 0`). Using a NOP here helps avoid unintended operation during or immediately after reset.

Function in the Pipeline

- **Maintains correct instruction flow** between the IF and ID stages by holding the fetched instruction and its PC value until the next clock, regardless of what changes in IF in the meantime.
- **Isolation and synchronization**: Ensures that staging between pipeline steps is clean, with transfers only on clock edges, making pipelined execution reliable and race-condition-free.
- **Resets to a safe state**, ensuring the pipeline doesn't propagate junk data after reset.

Summary Table

Condition	pc_out Value	instr_out Value
Reset (`rst=1`)	0	`0x0000_0013` (NOP)
Clock rising edge	`pc_in`	`instr_in`

In summary, this module acts as a crucial buffer between the fetch and decode stages, reliably passing on instructions while supporting clean system resets and pipeline operation.