Fetch（IF）模块功能描述和实现原理

在本处理器设计中，IF模块使用状态机接收flush信号，以此决定是否进行取指。通过接收重排序（ROB）、分支预测（BP）结果，以及指令长度所决定的默认指令地址，决定取指令的地址。IF模块与分支掩码（Branch mask）模块合作，确保分支指令的依赖关系，并记录需要回滚的指令，直到分支路径确定无误。最后将处理完成的指令地址发送给译码（ID）模块。

首先，IF模块依据（ROB>BP>default）的顺序，取得下条指令的值，将其存入寄存器。

随后，IF通过状态机判定是否需要对指令进行操作。依照逻辑：当接收到ROB的冲刷指令时，立刻固定至等待状态，否则当ID阶段准备好接受时，切换到运行状态。当分支单元（BU）提供有效的分支地址时，更新地址为分支目标，否则每周期顺序更新PC。

在处理BTB命中时，本处理器中的IF模块只接收第一条命中的指令，并用inst\_mask\_btb屏蔽后续指令。因为考虑即使当前IF返回了多条指令，但若第一条是跳转指令，后续指令都不会起效。在这一阶段，分支预测器的状态（GHR、是否跳转等）会被存入寄存器，并确保与memory对齐时序。在取指后，该pc前的指令会被屏蔽，避免误处理前一个fetch packet的旧指令。

再然后，IF模块会对所取指令的opcode进行初步处理，判断其是否为分支指令（bne,beq,etc..）或是jal与jalr。当所取指令是上述类别时，将跳转预测结果更新至GHR中，并将valid指示调整为1。

仅当当前fetch packet中所有有效的分支指令都能从mask freelist中申请到分支资源时，开始构建uop向量。IF模块通过遍历fetch packet中的每条指令，生成对应的uop，包含指令及其地址、分支预测结果，以及Branch mask的相关信息。在构建时，uop向量内的指令、指令地址取决于当前的PC地址。分支预测信息用BP模块提供的信息填充。Uop的有效性决定于指令本身的合法性、BTB是否屏蔽此条指令、core是否正在运行，以及（若是）分支指令是否分配了分支ID。

随后，IF模块与Branch mask模块配合，标记指令的控制流历史，以确保flush的时候不会出错。Branch mask模块会通知IF是否有Branch mask被释放，如有则把信息写入到指令uop中，这样做确保了当前指令不会重用提前被释放的branch id直到当前指令退休。

最终，IF将整个uop向量存入寄存器传给ID阶段进行后续译码操作，并保持在运行状态，等待下一组指令的处理。