第9.(1)找出数据依赖，①fld和②fdiv.d中f2有RAW冲突；④fld和⑤fadd中有RAW冲突.
⑥fadd和⑦fsd间有RAW冲突，⑨addi和⑪sub间有RAW冲突，⑪sub和⑫bne间有RAW冲突.；⑧与⑨有RAW冲突.

∴指令①从第1个周期被执行.至第4个周期执行完毕；指令②从第5个周期开始执行，至第15个周期完成

指令③：第6个周期开始，第10个周期结束；指令④：第7个周期开始，第10个周期结束.

指令⑤：第11个周期开始，第13个周期结束；指令⑥：第16个周期开始，第18个周期结束(与②有冲突).

指令⑦：第19个周期开始，第20个周期结束；指令⑧：第20个周期开始，第21个周期结束

指令⑨：第21个周期开始，第21个周期结束 指令⑩：第22个周期开始，第22个周期结束

指令⑪：第23个周期开始，第23个周期结束 指令⑫：第24个周期开始，第25个周期结束 ∴共用25个周期

(2)(在第2和3问中，简略汉字，用xx~xx替代)：

指令①：1~4 指令②：5~15 指令③：5~9 指令④：6~9 指令⑤：10~13 指令⑥：16~18.

指令⑦：19~20 指令⑧：19~20 指令⑨：20~20 指令⑩：20~20 指令⑪：21~21 指令⑫：22~23

∴共需要23个周期完成.

(3)由于在实际情况中，①②、④⑤、⑨⑩、⑥⑦、⑪⑫之间出现真数据冲突，将指令⑦和⑧交换

指令①：1~4 指令②：5~15 指令③：5~9 指令④：6~9 指令⑤：10~13 指令⑥：16~18.

指令⑧：16~17 指令⑦：19~20 指令⑨：19~19 指令⑩：20~20 指令⑪：20~20 指令⑫：21~22

∴共需要22个周期完成，可加快一个周期(同样，将指令⑧~⑪指到最前也可节省1个周期，但两者结合由于双发射限制仍只能省1个周期，故不列举) 序列：①②④⑤⑥⑧⑦⑨⑩⑪⑫(一种可能)

10:开始:loop: fld f4,0(a_0)	重命名后: fld T_0, 0(a_0)
fmul.d f_2, f_0, f_2	fmul.d T_{10}, T_0, T_2
fdiv.d f_8, f_4, f_2	fdiv.d T_{11}, T_9, T_{10}
fld f4,0(a_1)	fld T_{12}, 0(a_1)
fadd.d f_6, f_0, f_4	fadd.d T_{13}, T_0, T_{12}
fsub.d f_8, f_8, f_6	fsub.d T_{14}, T_{11}, T_{13}
fsd f8,0(a_1)	fsd T_{14}, 0(a_1)

11.区别：显式重命名：ROB不记录指令的结果，即将提交的数据和处于推测状态的数据都保存在物理寄存器中，故物理寄存器数目要高于逻辑寄存器数目。

隐式重命名：ROB保存正在执行、尚未提交的指令的结果，ARF(ISA 规定的 register file)保存已经提交的指令中即将写入寄存器的值。ARF只保存已经提交的指令的值，处于推测状态的值由ROB保存，故物理寄存器数量与逻辑寄存器数量相同。另外，还需建立一个映射表，记录操作数在ROB的位置。

优缺点：相比于显式重命名，隐式重命名需要的物理寄存器数目更少，但是每个操作数均在其生命周期中需要保存在ROB和ARF两个位置，读取数据的复杂度较高，功耗更高。

实现方式：显式重命名：map_table：记录逻辑寄存器与物理寄存器间的映射关系；free_list：记录物理寄存器空闲状态；busy_table：记录寄存器是否可读，ROB记录提交的数据。先通过map_table获取源操作数对应的物理寄存器，然后由free_list动态分配一个空闲物理寄存器作为目的寄存器，最后由busy_table判断寄存器是否可读，可读指令就发射。

隐式重命名：主要由ROB保存正在执行且尚未提交指令的结果，而ARF保存已经提交的指令中即将写入寄存器中的值，同时通过一个新建立的映射表，记录操作数在ROB中的位置，同时记录对应寄存器的最新值是保存在ROB还是ARF中。指令提交是由ROB交给ARF，而指令被写入ROB便认为完成重命名。