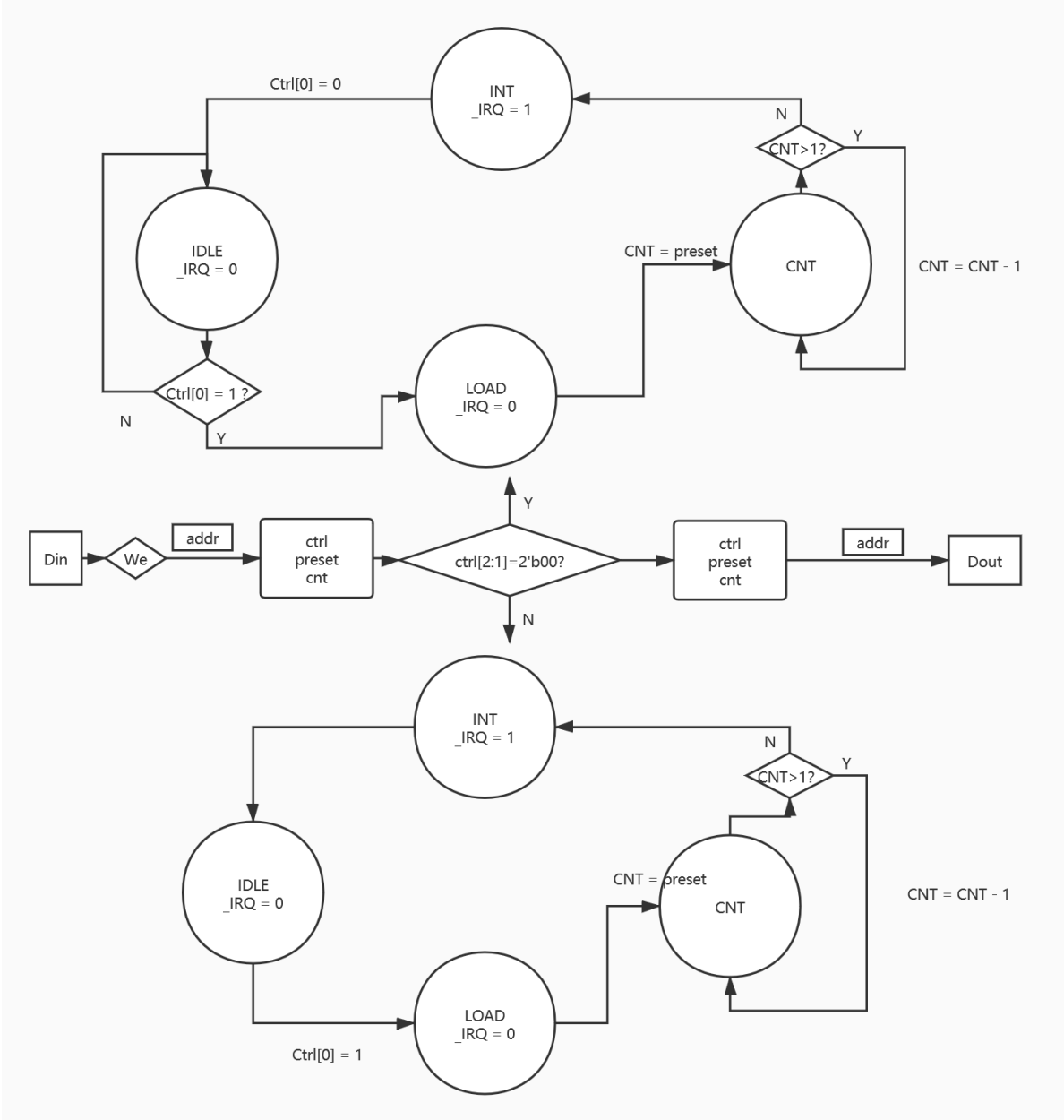
P7计时器说明

1.状态图及说明

定时器的状态图如图所示：



如图所示，该定时器有两种工作状态

1. 当ctrl[2:1]!=2'b00时，在首次将ctrl[0]设置为1即启动定时器后，定时器将循环工作，每次在cnt=1时进入interrupt状态，将_IRQ信号置1，下一个时钟周期将进入IDLE状态，除非外部更改控制条件，该模式下定时器将产生脉冲式的终端信号。
2. 当ctrl[2:1]=2'b00时，通过将ctrl[0]设置为1启动定时器工作，在cnt从preset递减到1时产生interrupt信号，此后将在IDLE状态下等待外部控制将ctrl[0]再次置1后才会继续工作，即定时产生1次中断信号。

2.计时器使用说明

通过向计时器的三个寄存器写入值来实现对计时器的控制，具体而言：

ctrl（7F00\7F10）：

3	2-1	0
中断请求掩码位	计时器模式（00时定时产生一次中断，其他信号时产生等周期的中断脉冲）	启动位（为1时启动定时器）

present（7F04\7F14）：

31-0
产生中断的延迟\周期，经过present个时钟周期产生中断

cnt（7F08\7F18）：

31-0
计时器当前计数值

在实际使用定时器时，这三个寄存器仅有cnt与ctrl[0]会由计时器主动变化，其他为将保证为最近一次CPU对其的更新不变。

操作规范

1. 仅通过lw、sw与定时器交互，不应该向cnt寄存器写入值
2. 仅在IDLE阶段时，可以向present寄存器和ctrl寄存器写入值

误操作的后果：

1.在load阶段时更新present寄存器，此时向cnt写入的初值将有可能在其时钟边沿附近不稳定而变的无法确定。

2.在init阶段更新ctrl：产生的中断请求信号可能出现毛刺（更新ctrl[3]时产生）导致CPU接受到错误的中断请求信号；而对ctrl[2:1]的更新可能导致计时器对其模式判断变的　不确定而导致自主更新的ctr[0]也变的不确定；而对ctrl[0]的更新可能与计时器对ctrl[0]的自主更新信号产生冲突从而变的无法确定。

3.向cnt寄存器写入值：在非cnt模式下，对于cnt的更新没有意义，而在cnt模式下写入，会导致cnt-1信号与更新的cnt信号相互竞争，具体值因而变的无法确定。

注：这里的无法确定指的并不是变为高阻态Z，而是其具体值确定但取决于各部分信号产生的延迟和时钟沿处对信号稳定的判定，即具体值取决于具体的电路实现。