电子信息与电气工程学院

第七届青年教师教学竞赛

 教学设计（范例）

**《VLSI数字通信原理与设计》**

课时4：重定时技术

|  |  |
| --- | --- |
| 授课题目 | 重定时技术（选自教学内容第三章：重定时） |
| 学情分析 | 一、学生知识背景学生在之前已修“计算机体系结构”和“数字逻辑”，对流水线技术有所了解。二、学生特点分析本节课授课对象为微电子本科三年级学生，具有一定专业基础，可以很快理解与掌握知识点，但自主学习的能力与灵活运用原理的能力不足，需要教师实际引导。三、教学内容分析本次教学内容为重定时的基本概念和作用，引导学生能够运用k倍降速的割集重定时方法，提升系统的处理能力。在本次授课过程中，采取将教师与学生结合起来的教学方式，教师讲授完知识点后，由学生就相关知识点发表自身的看法，增加了学生在课堂上的参与度。学生根据课前相关文献的阅读、对重定时有一定了解后，可为教师授课奠定良好的基础。培养学生独立发现问题、分析问题、解决问题的能力。本节内容难度适中。 |
| 课程思政 | 结合本次课的教学重点，以实际某图像处理芯片设计为例，介绍其探索重定时技术降低设计周期，从而节省成本，培养学生追求卓越、精益求精的工匠精神，并进一步增强对本专业的学习兴趣，理解EDA公司将重定时技术集成在设计工具中的动机。 |
| 教学目标 | 1. 认知目标

学生能够描述重定时的基本概念和作用，区分割集重定时、节点重定时和流水线重定时。1. 能力培养目标

学生能够运用k倍降速的割集重定时方法，提升系统的处理能力。学生能够针对不同的目标应用采用不同的解决方法。当芯片设计完成后、如果实际的工作频率距离目标还有少量差距，若重新再做可能要花费数个月的时间，此时可以利用重定时的手段，通过平衡每一段路径的延迟，少量地提高系统工作频率。 |
| 教学思想 | 1. 鼓励学生运用相关技术进行数字信号处理算法VLSI设计，帮助他们将所学知识熟练地应用于学习生活中去。
2. 引导学生独立发现、分析和解决问题。
3. 训练学生清晰思考和语言文字准确表达的能力，提高学生对知识掌握的能力。
 |
| 教学内容 | 1. 重定时的基本概念。
2. 重定时的方法。
3. 重定时的作用。
4. 重定时的应用。
 |
| 教学重点 | 重定时的方法：割集重定时、节点重定时、流水线重定时、k倍降速的割集重定时 |
| 教学难点 | 重定时的数学定义。 |
| 教学方法 | 1. 文献指导法：在授课之前，给学生推荐与重定时有关的书籍或者期刊论文，让学生对重定时有一定的了解。
2. 小组讨论法：给学生提问：既然有了流水线，为什么还要学习重定时技术？根据课前推荐的相关文献和课堂的授课内容，让学生分组讨论，引导学生积极思考问题。
3. 案例分析法：分析与重定时有关的典型案例，给学生重点讲解难点，指导学生更好地运用知识点解决问题。
 |
| 教学策略 | 1. 提前上传课件，让同学提前了解本节课程基本内容；并推荐相关文献，供学生课前阅读。
2. 以平衡速度、面积和功耗为驱动，引入重定时的应用。
3. 设置学生汇报环节，让学生可以对相关知识点发表自己的观点，使学生成为课堂的中心。
4. 针对教学难点，首先阐述重定时的数学定义的意义在于，面对大规模的电路时，可以通过把它转成数学模型，通过数学的方式来做寄存器的权衡，从而利用工具来自动实现重定时。然后配合PPT动画和具体例题来对重定时的数学定义进行详细讲解。
 |
| 教学过程 |
| 教学要旨：为了达到更快的运行速度，可以在关键路径插入流水线寄存器，所付出的主要代价就是面积变大。相反，在有些情况下不需要那么快的运行速度，而是想减小面积，可能需要撤去某些流水线，以减少延时的数目。为了能在各个性能指标之间进行灵活的折中，就希望能制定一套增加或减少系统延时数目以及改变系统延时分布的方法，重定时技术就应运而生了。首先，介绍重定时技术的基本概念，通过实例的讲解，介绍重定时的基本规则及其性质。然后引出两种重定时：割集重定时和流水线重定时。接着通过具体例子详细讲解割集重定时的过程，借助数据流图来帮助学生形象理解割集重定时的过程，并辅助板书，清晰展示教学思路。然后引出割集重定时的一个特例：节点重定时。同样，通过数据流图讲解其过程，引导学生自己思考，把前面割集重定时的思路举一反三，巩固学习到的知识。接着引出重定时在流水线上的应用：前馈割集重定时，再扩展到k倍降速后的割集重定时。通过一个三阶格型滤波器的例子，加深对重定时的理解，掌握重定时的方法。然后讲解重定时的两个作用：加快速度和减小面积。通过一个简单数据流图的例子，形象地说明重定时的作用，使学生对重定时能够全面认知。接着讲解两个课后习题，巩固割集重定时和k倍降速重定时，通过提问，了解学生的掌握情况，对学生掌握不够的知识点着重讲解。最后对本讲内容进行概括性总结。 |
| 教学安排 | 1. 重定时的背景介绍（3分钟）
2. 首先提出一个芯片制造中的常见问题：当电路设计完成后，发现时钟频率差一点达到要求，那么该如何解决？
3. 然后通过回顾流水线技术，提出流水线技术会带来面积增大的问题，从而引出重定时技术能够通过减少关键路径和寄存器数目来减少面积。
4. 重定时的基本概念（7分钟）
5. 重定时技术的定义：是一种变换技术，在不改变系统的输入输出特性的前提下，改变电路延迟元件的配置。
6. 延时单元可以在所有输出与所有输入之间移动，主要方法是减少关键路径与寄存器数量。
7. 最后给出一个三级流水线的具体实例帮助学生理解重定时如何改变关键路径和寄存器数目。
8. 重定时的分类（13分钟）
9. 最基本的重定时是割集重定时，给学生讲解割集重定时，并用图表法进行展示：

具体方法：* 在一个方向的边上增加延时。
* 在另外方向的边上减少同样的延时。
1. 用框图强调节点重定时和流水线是割集重定时的特例。

节点重定时流水线重定时1. 扩展到k倍降速后的割集重定时。通过一个三阶格型滤波器的例子，加深对重

定时的理解，掌握重定时的方法。1. 重定时的数学定义和性质（20分钟）
2. 重定时的数学定义
* 用有向图*G*表示电路：

顶点𝑣表示组合逻辑单元，顶点值表示组合逻辑的延迟。边𝑒表示两组合逻辑间的连接，权重𝑤表示边上的寄存器数。* 重定时：将一个电路*G*映射到一个重定时电路*G*r ，即*G* →*G*r。
* 重定时值：每个节点*V*的重定时值*r*(*V*) 用来表征*G* → *G*r的重定时解。
* 重定时方程：用来确定节点*U*到*V*的边*U* → *V*重定时后的权重。即：
* 路径权重*W*：从顶点𝑢到𝑣任一路径的最小寄存器数。
* 路径延迟*D*：从顶点𝑢到𝑣任一包含*W*路径的最大延迟。
1. 重定时的算法

已知：𝑤(𝑒), 𝑊(𝑢,𝑣), 𝐷(𝑢,𝑣)，目标时钟周期𝑇目标：求一组重定时值约束：* 重定时后任意边的寄存器不小于0，即
* 对于路径延迟𝑫(𝒖,𝒗)>𝑻的边，关键路径上至少有一个寄存器，即

 给出重定时算法的流程图如下：1. 用IIR滤波器为例，详细讲解割集重定时的过程，使学生可以阐述重定时方程的作用。同时给出重定时的实际应用，以Xilinx Vivado 2016.3为例，介绍如何使用工具进行重定时。

1. 根据重定时方程归纳重定时性质
* 对于重定时的路径p= V0 → V1 → … → Vk，只与路径起始、终止节点的重定时值相关。
* 重定时不改变环路中的总延迟数：因为环路的Vk=V0
* 重定时不改变DFG的迭代边界T∞：T∞=TL/W ，因为环路的运行时间和延迟数都不变化。
* 所有节点重定时值r(V)都增加常数值j，重定时映射G → Gr不变。
1. 课堂总结（2分钟）

总结本节课程内容，布置作业，巩固割集重定时和k倍降速重定时。 |
| 教学流程设计 | 上一讲内容回顾本章总结，布置作业重定时的重要作用是否割集重定时节点重定时举例重定时的算法学生是否掌握重定时的基本概念流水线重定时K倍降速后的割集重定时本章知识框架及学习目标介绍 重定时的数学定义  |
| 板书设计 |  |
| 课程作业 | 作业内容：教材习题1、习题11作业设计思路：习题1考察迭代边界的计算方法、关键路径的计算和流水线重定时的方法；习题11考察流水线重定时的方法以及k倍降速后的重定时方法。 |
| 思考题 | 割集重定时中常用的技术是倍降速。倍降速在重定时前把每个寄存器换为一组个寄存器，从而方便后续重定时优化。通过阅读文献回答倍降速的优点和缺点是什么？ |
| 参考文献 | N. Weaver, Y. Markovskiy, Y. Patel, J. Wawrzynek. “Postplacement C-slow retiming for the Xilinx-Virtex FPGA”. Eleventh ACM International Symposium on Field Programmable Gate Arrays, 2003: 185-194. |
| 教学评价 | 对教学过程的评价：1. 通过观察学生课堂中的专注度，以及对“割集重定时、节点重定时和流水线重定时有什么区别”等问题的反映，判断学生在本节课教学过程中的投入程度。
2. 通过与学生的眼神交流和互动情况，判断本节教学设计内容组织和节奏控制的合理性。

对教学目标达成度的评价：1. 通过学生在课堂上以小组工作方式完成割集重定时的课堂练习并进行展示，根据实际完成质量来判断学生对本节知识点的理解程度和应用能力。
2. 根据学生课后作业的完成情况来判断学生对知识点的掌握情况。
 |
| **本章重点词汇**1. 重定时（Retiming）
2. 割集重定时（Cutset Retiming）
3. 节点重定时（Node Retiming）
 |