Temporal Placement

HV: Hồ Hoàng Nguyên
GVHD: TS. Trần Ngọc Thịnh

Reconfigurable Computing

Outline

1. **Offline Temporal Placement**
   - First-Fit and Best-Fit Placement
   - Packing Approach for Temporal Placement
2. **Online Temporal Placement**
   - Managing the Device's Free Space with Empty Rectangles
   - Managing the Device's Occupied Space

Wasted Resources

- **Wasted resource $wr(v_i)$ of a node v_i:**
  - Unused area occupied by the node v_i during the computation of a partition.
    $wr(v_i) = (t(P_i) - t_i) \times a_i$,
    $t(P_i)$: run-time of partition P_i.
    t_i: run-time of the component v_i
    a_i: area of v_i
- **Wasted resource avoidance:**
  - A component is placed on the chip only when its computation is required
  - and remains on the device only for time it is active.
    - → Idle components can be replaced by new ones.

  ⇒ Partial reconfiguration
- **Assumptions:**
  - There is partial reconfiguration support.
  - Blocks are hard

3

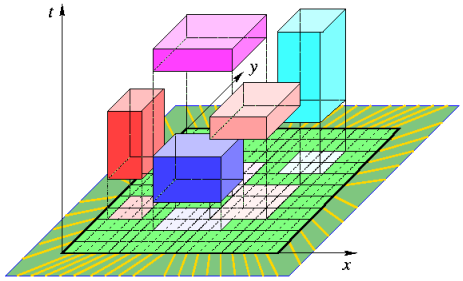
Temporal placement -Motivation

- **Temporal partitioning**
  - Advantages:
    - Off-line computation
    - Technically easy to implement
    - Can be implemented on each reconfigurable device
  - Drawbacks:
    - Lack of dynamics
    - Resource wasting
- **Temporal placement**
  - Reduce the amount of wasted resources
  - Dynamic decision-making
  - Two approaches
    - Off-line (pre-defined placement sequence)
    - On-line (dynamic placement sequence)

4

Temporal Placement

- **Temporal placement:**
  - Time-dependent placement
  - Management of task execution at run-time
- **Graphical Representation:**
  - t axis: life time of a configuration.
  - Some overlaps (resource sharing) in 2-D is allowed:
    - If not at the same time.
  - A horizontal cut: configuration at a given time



5

Temporal Placement

- **Off-line (compile time)**
  - Pre-defined placement sequence
    - In DSP applications, program flow can be predicted.
- **On-line (during execution)**
  - Computation sequence is not known in advance.
    - → Dynamically at run time.
  - Needs to solve computational intensive problems in a fraction of millisecond:
    - Efficient management of the free space
    - Selection of the best site for a new component
    - Management of communication

6

Outline

1. **Offline Temporal Placement**
   - First-Fit and Best-Fit Placement
   - Packing Approach for Temporal Placement

7

Off-Line Temporal Placement

- **Off-line temporal placement:**
  - Given a DFG $G = (V,E)$ and a reconfigurable device H (with length H_x and width H_y), a *temporal placement* is a 3-dimensional vector function:

    $p = (p_x, p_y, p_t) : V \rightarrow \mathbb{R}^3$,

  p_t defines a feasible schedule ($p_t(v_i)$: Starting time of v_i)

  $p_x(v_i)$, $p_y(v_i)$: Coordinates of v_i on H,

8

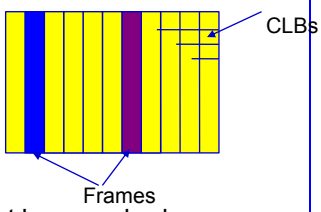
First/Best Fit Algorithm

- **First fit:**
  1. Select a node among ready nodes.
     - i.e. nodes whose predecessors have been placed
  2. Place it in the first free location.
- **Advantages:**
  - Fast (linear w.r.t. number of free locations)
- **Disadvantages:**
  - Unused resources very high
  - Fragmentation

9

First/Best Fit Algorithm

- **Best fit:**
  1. Select a node among ready nodes.
  2. Place it in the best free location.
- **Advantages:**
  - Better area utilization
- **Disadvantages:**
  - Much slower:
    - Complete list of free locations must be searched.
  - Fragmentation
- **Simplifying the problem:**
  - Restricting the modules to columns (Virtex II) or part of a column (Virtex 4 / Virtex 5 / Virtex 6).



10

First Fit Algorithm

- **Algorithm:** First-fit temporal placement of clusters

1. While all the clusters are not placed do

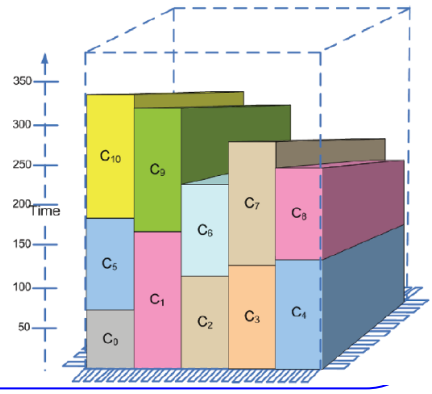
1.1. Select one cluster C_{act} from the list of ready-to-run clusters

1.2. From the clusters already placed, select the cluster C_{top} with the smallest finishing-time such that

$C_{top} \leq C_{act}$

1.3. Place the cluster C_{act} on top of C_{top}

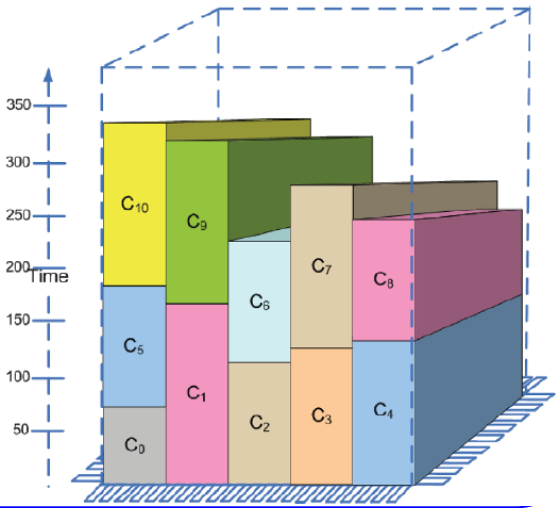
2. end while



11

First Fit Algorithm

- **Configuration sequence produced:**
  - **a) {0, 1, 2, 3, 4}**
  - **b) {5, 1, 2, 3, 4}**
  - **c) {5, 1 , 6, 7, 4}**
  - **d) {5, 1 , 6, 7, 8}**
  - **e) {5, 9 , 6, 7, 8}**
  - **f) {10, 9 , 6, 7, 8}**



12

Packing Approach

- **Variations of Packing Problem:**

1. **Base Minimization Problem (BMP):**
   - Given a set of boxes *B* and a height *H*, find a container with minimal size (*x, y, H*) that can accommodate the set of boxes *B*
   - i.e. in temporal placement:
     - Find the device with minimum size (*x, y*) on which a set of components can be implemented given an overall run-time $t = H$.
2. **Strip Packing Problem (SPP):**
   - Given a set of boxes and a base (*X, Y*), find the minimum height *h* container with size (*X, Y, h*) that can hold all the boxes.
   - i.e. in temporal
     - Find the minimum run-time $t = h$ for a set of components given a reconfigurable device with size *(X, Y)*.

- **The device is usually fixed**
  - → SPP is more common.

13

Packing Classes

- **Interval Graph:**
  - Given a DFG $G = (V,E)$, a reconfigurable device *H* and a packing of *V* into *H, (*i.e. a 3-D placement of the components of *V* on the device *H),* an *interval graph* of *G* is a graph $G_i = (V,E_i)$, $\forall i \in \{1, 2, 3\}$ such that:

  $(v_k, v_l) \in E_i \Leftrightarrow$ the projections of the nodes v_k and v_l overlap in the *i*-th dimension.
- **Complement Graph:**
  - Complement of interval graph:

    $\overline{G_i} = (V,\overline{E_i})$

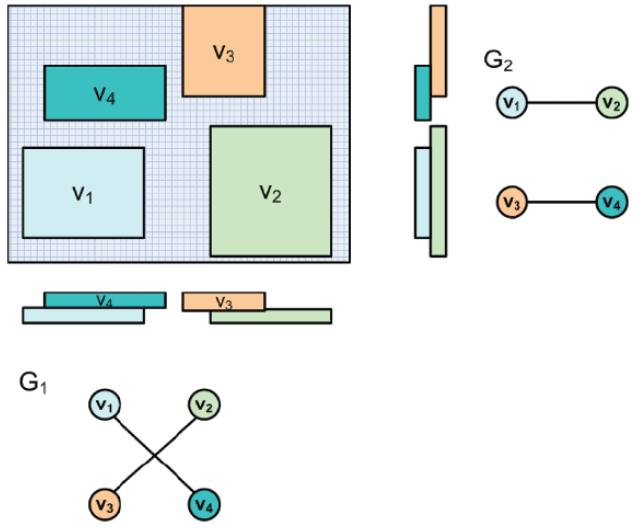
14

Packing Class

- **Packing class:**
  - A d-tuple of interval graphs with the following properties:
    - **C1:** Any independent set $S \subseteq G_i$ is i-admissible
      $\sum_{v \in S} w_i(v) \leq h_i$) (each set of boxes must fit in the container in the i-th direction)
      w_i = length of projection of v_i in i-th direction
      h_i is the length of the device in the same dimension.
    - **C2:** $\cap_{i=1...d} E_i = \varnothing$. There must be at least one dimension in which the boxes do not overlap.
- The G_i are called ***component graphs*** of $E = \{E_1, E_2, E_3\}$.

15

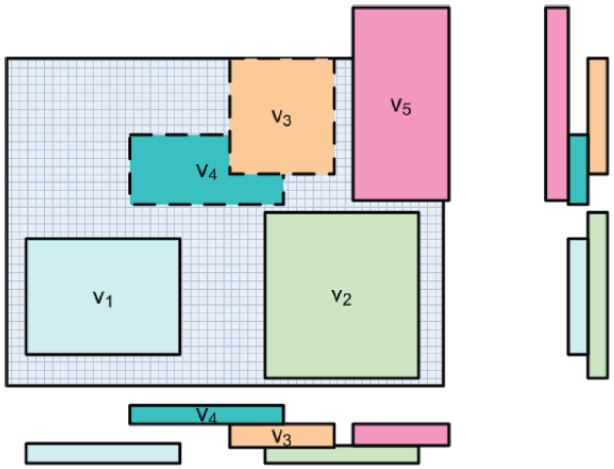
Packing Class



16

Packing Class

- Invalid two-dimensional packing:



17

Packing Class

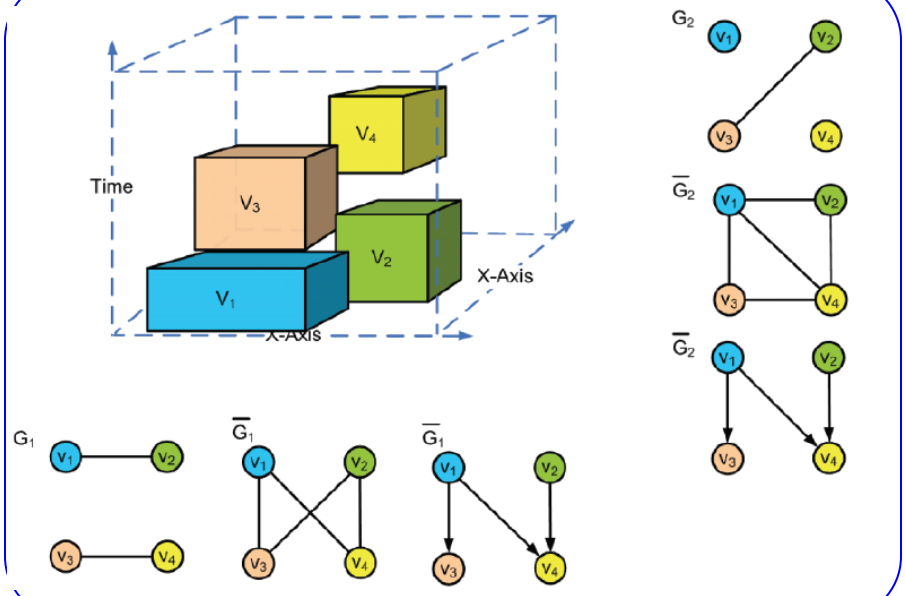
- **Theorem:**
  - A d-tuple of graphs $G_i(V_i,E_i)$ corresponds to a feasible packing, if and only if it is a packing class.
- **Comparability Graph:**
  - ***Given a DFG G = (V,E), a reconfigurable device H and a packing of V into H, The comparability graph of an interval graph $G_i = (V,E_i)$, $\forall i \in \{1, 2\}$ is the directed graph $G_i = (V,E_i)$, where $(v_k, v_l) \in E_i \Leftrightarrow v_l$ is 'placed after' v_k in 3-rd dimension, i.e. $(p_t(v_k) + t_k \leq p_t(v_l))$.***
    - The relation *'place after'* defined by the comparability graph is a transitive relation also known as *transitive orientation* that can be used for orienting the packing classes.

18

Packing Class Orientation

- **Packing Class Orientation:**
  - Given G(V, E), H and packing of V on H, *orientation* of the packing class corresponding to the packing is defined by constructing the comparability graph of the interval graph in the time dimension.

19



20

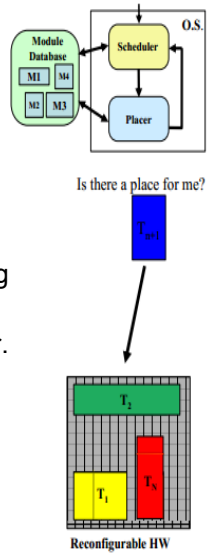
Outline

1. **Offline Temporal Placement**
   - First-Fit and Best-Fit Placement
   - Packing Approach for Temporal Placement
2. **Online Temporal Placement**
   - Managing the Device's Free Space with Empty Rectangles
   - Managing the Device's Occupied Space

21

Online Temporal Placement

- **Possible scenario**
  1. The scheduler requests the placement of a new task
  2. The placer tries to find a place on the device for the new task
     1. Upon success, it acknowledges the scheduler and downloads the corresponding module to the device. Fine!
     2. Upon failure, it acknowledges the scheduler. The scheduler can now decide to reject the request or to try it later again
  3. In all cases, we have to decide on-line, if a new component can be placed on a device where other components are already running



22

Online Temporal Placement

- **Definition:**

  Given a reconfigurable processing unit H at time t with a given configuration Ct.

  Find an optimal position for a new incoming task v such that v does not overlap with any running component in the configuration Ct

23

Outline

1. **Offline Temporal Placement**
   - First-Fit and Best-Fit Placement
   - Packing Approach for Temporal Placement
2. **Online Temporal Placement**
   - Managing the Device's Free Space with Empty Rectangles
     - KAMER
   - Managing the Device's Occupied Space

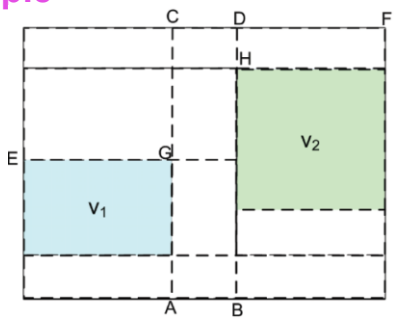
24

KAMER

- **Empty Rectangle (ER):**
  - *A rectangle that does not overlap a placed module on the chip*
- **Maximum Empty Rectangle (MER):**
  - *An ER not included in any other rectangle than the device bounding box*

25

Example



- **Non-ER:**
  - (E,F)
- **ER:**
  - (E,D): MER
  - (A,D): MER
  - (E,C): not MER
- **KAMER method:**
  - *keeping all maximum empty rectangles:*
    - Permanently keeps track of all MERs
    - Whenever a request for placing a component *v* arrives, the list of MERs is searched for a rectangle that can accommodate *v*.
  - Possible to have many MERs in which *v* can fit

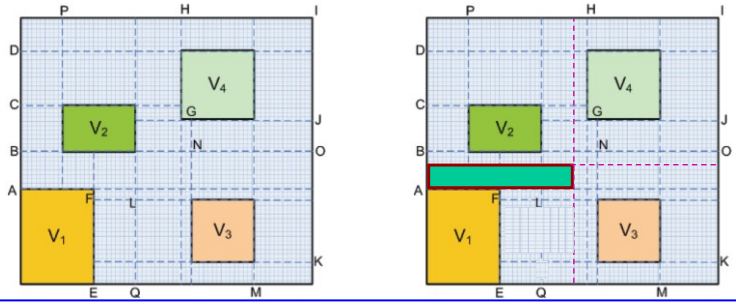
  => Strategies: first-fit or best-fit

26

KAMER

- Once a rectangle is chosen:
  - Candidate points are those that do not allow an overlap with the external part of the rectangle.

- **Problem 1:**
  - Number of empty rectangles does not grow linear with the number of components included



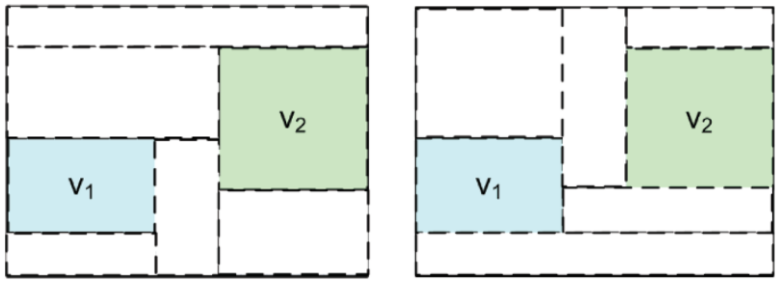
27

KAMER

- **Run-time of free rectangle placement: $O(n^2)$**

- Heuristic:
  - Keep only the *non-overlapping empty rectangles*
    - → Linear time, lower quality

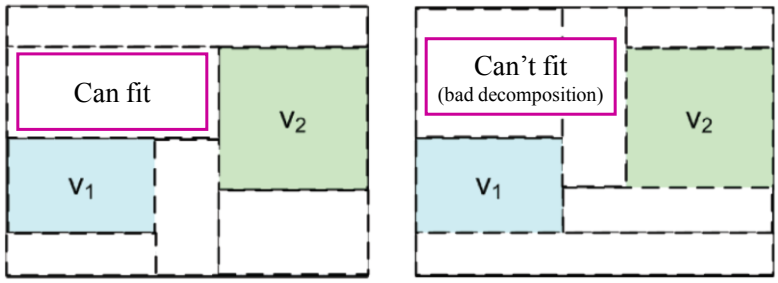
- **Problem 2:**
  - Several non-overlapping rectangle representations



28

Keeping Non-Overlapping ERs

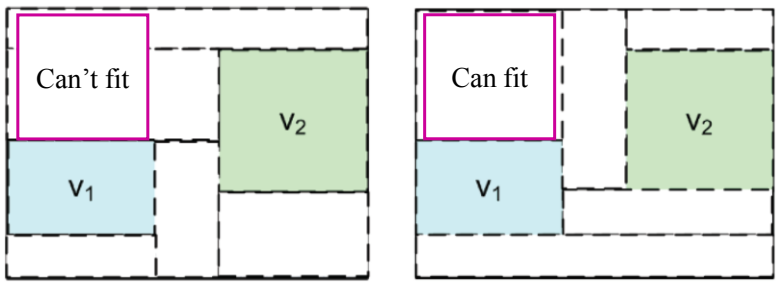
- **Problem 3:**
  - Non-overlapping empty rectangles are not necessarily maximal
    - → A module may exists that could fit onto the device, but cannot be placed



29

Keeping Non-Overlapping ERs

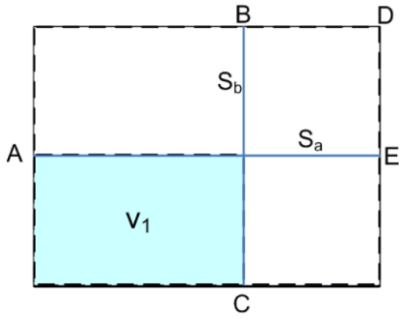
- **Problem 2:**
  - Non-overlapping empty rectangles are not necessarily maximal
    - → A module may exists that could fit onto the device, but cannot be placed



30

Keeping Non-Overlapping ERs

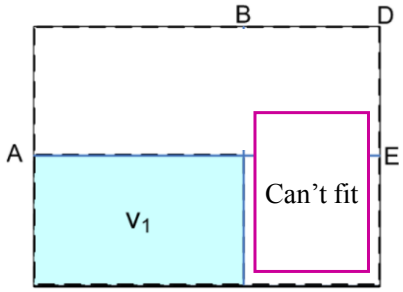
- **Incremental update:**
  - Whenever a new component *v*1 is placed in a rectangle, two possibilities:
    - Horizontal splitting
    - Vertical splitting
  - Negative impact on next components



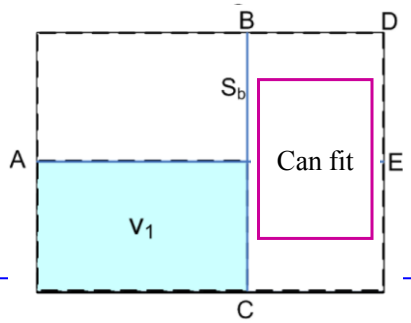
31

Keeping Non-Overlapping ERs

- **Horizontal**



- **Vertical**



- **Solution:**
  - Delaying the split decision for a number of steps later

32

Outline

1. **Offline Temporal Placement**
   - First-Fit and Best-Fit Placement
   - Packing Approach for Temporal Placement
2. **Online Temporal Placement**
   - Managing the Device's Free Space with Empty Rectangles
   - Managing the Device's Occupied Space

33

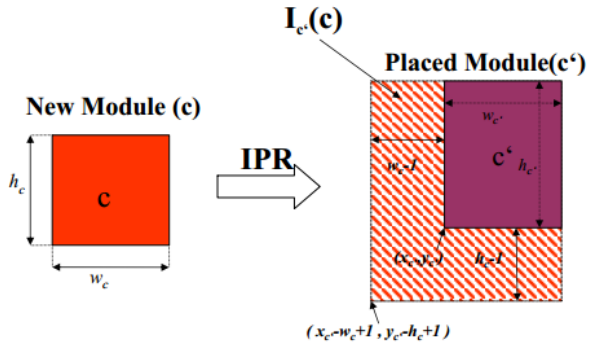
Managing the Device's Occupied Space

- **Observation:**
  - The number of free rectangles increases much faster than the number of occupied rectangles.
- **Method:**
  - Instead of managing the empty space, we manage the occupied space

34

Managing the Device's Occupied Space

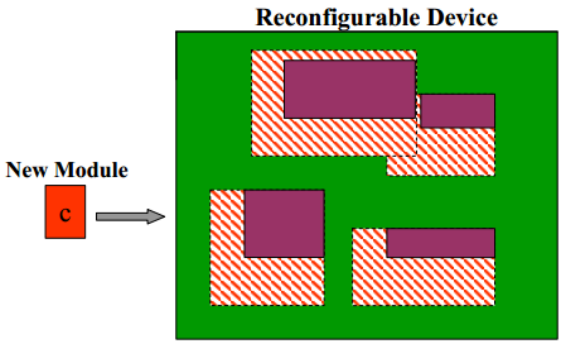
- **Definition :**
  - The impossible placement region of a module C relative to a running module C', $I_{c'}$ (c) is the area where C cannot be placed without overlapping with C'.



35

Managing the Device's Occupied Space

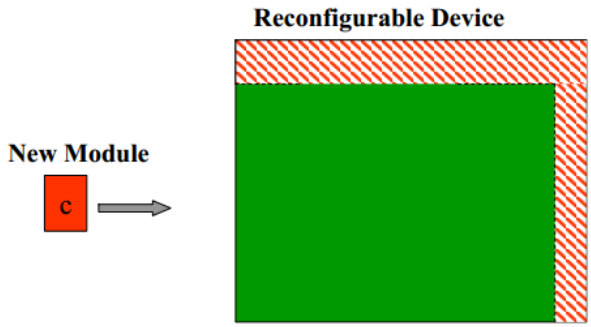
- **Definition :**
  - The impossible placement region of a module C relative to a set S of running modules is the union of the impossible placement regions of C relative to each module in S.



36

Managing the Device's Occupied Space

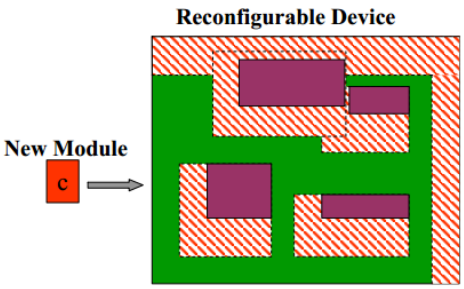
- **Definition :**
  - The impossible placement region of a module C relative to the device is the region where the module cannot be placed without overlapping with the border of the device.



37

Managing the Device's Occupied Space

- **Definition:**
  - The impossible placement region of a module C is the union of all impossible placement regions
- **C may be placed everywhere in the complement of the impossible placement region.**



38

Summary

- **Temporal placement**
  - Offline Temporal Placement
    - First-Fit and Best-Fit Placement
    - Packing Approach for Temporal Placement
  - Online Temporal Placement
    - Managing the Device's Free Space with Empty Rectangles
      - KAMER
    - Managing the Device's Occupied Space
  - Advantages:
    - Highly flexible and efficient in terms of device utilization
  - Disadvantages:
    - Algorithms are difficult and cost-intensive

39

Thank you

40