CPEG 422/622 EMBEDDED SYSTEMS DESIGN

MW 3:35 – 4:50, SHARP 116

Chengmo Yang

chengmo@udel.edu

Evans 201C

LECTURE 14
HW/SW CO-DESIGN

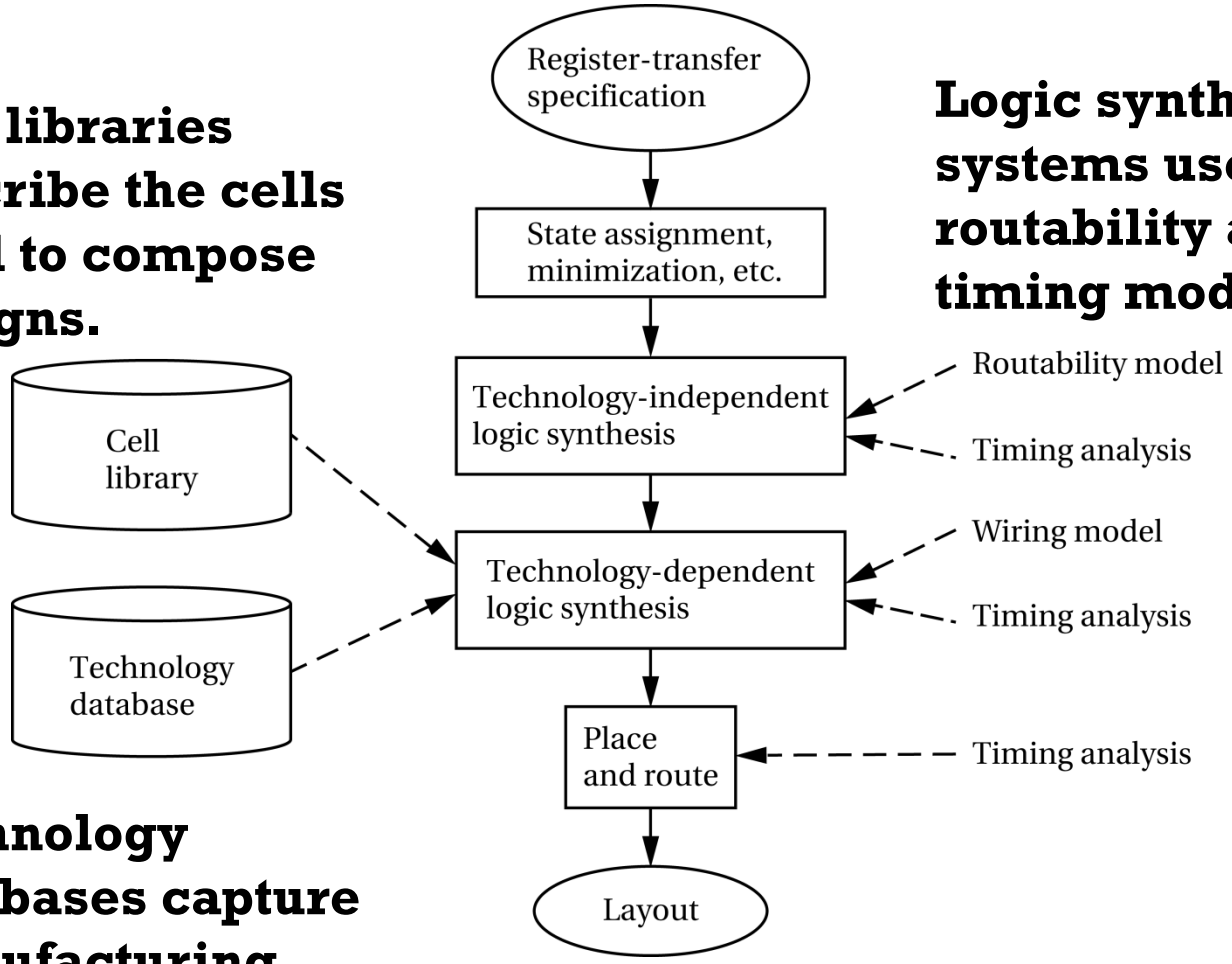
OUTLINE

- Various design flows
  - Hardware design flow
  - Platform-based design
  - HW/SW co-design flow
- Partitioning
- Scheduling

HARDWARE DESIGN FLOW

Cell libraries describe the cells used to compose designs.

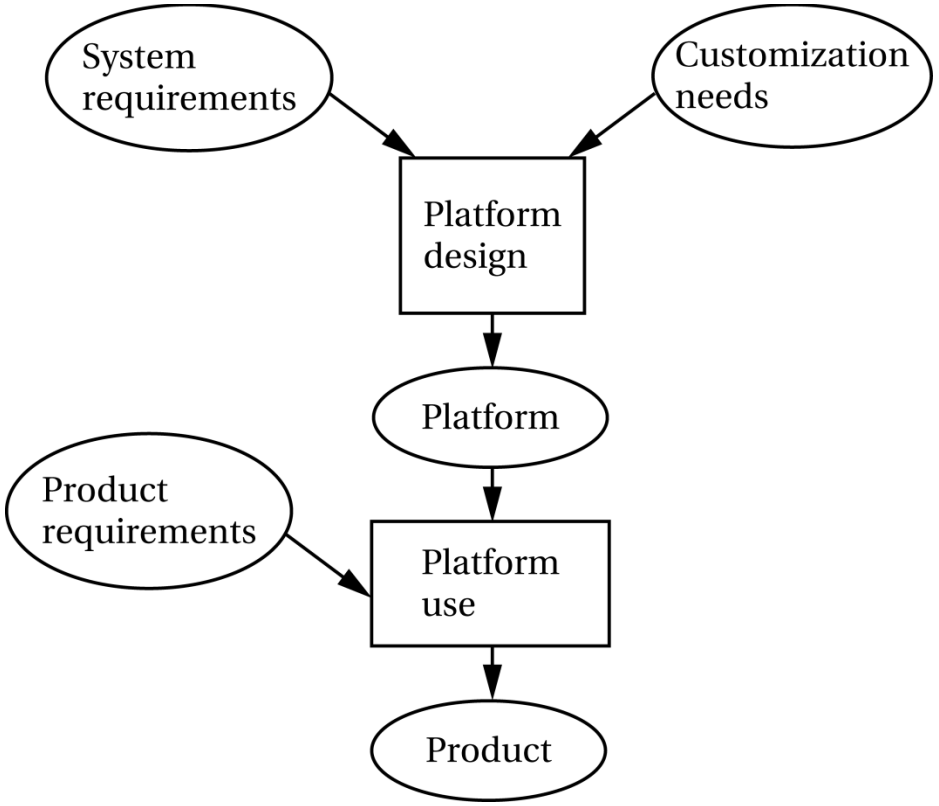
Logic synthesis systems use routability and timing models.



Technology databases capture manufacturing process information.

PLATFORM-BASED DESIGN

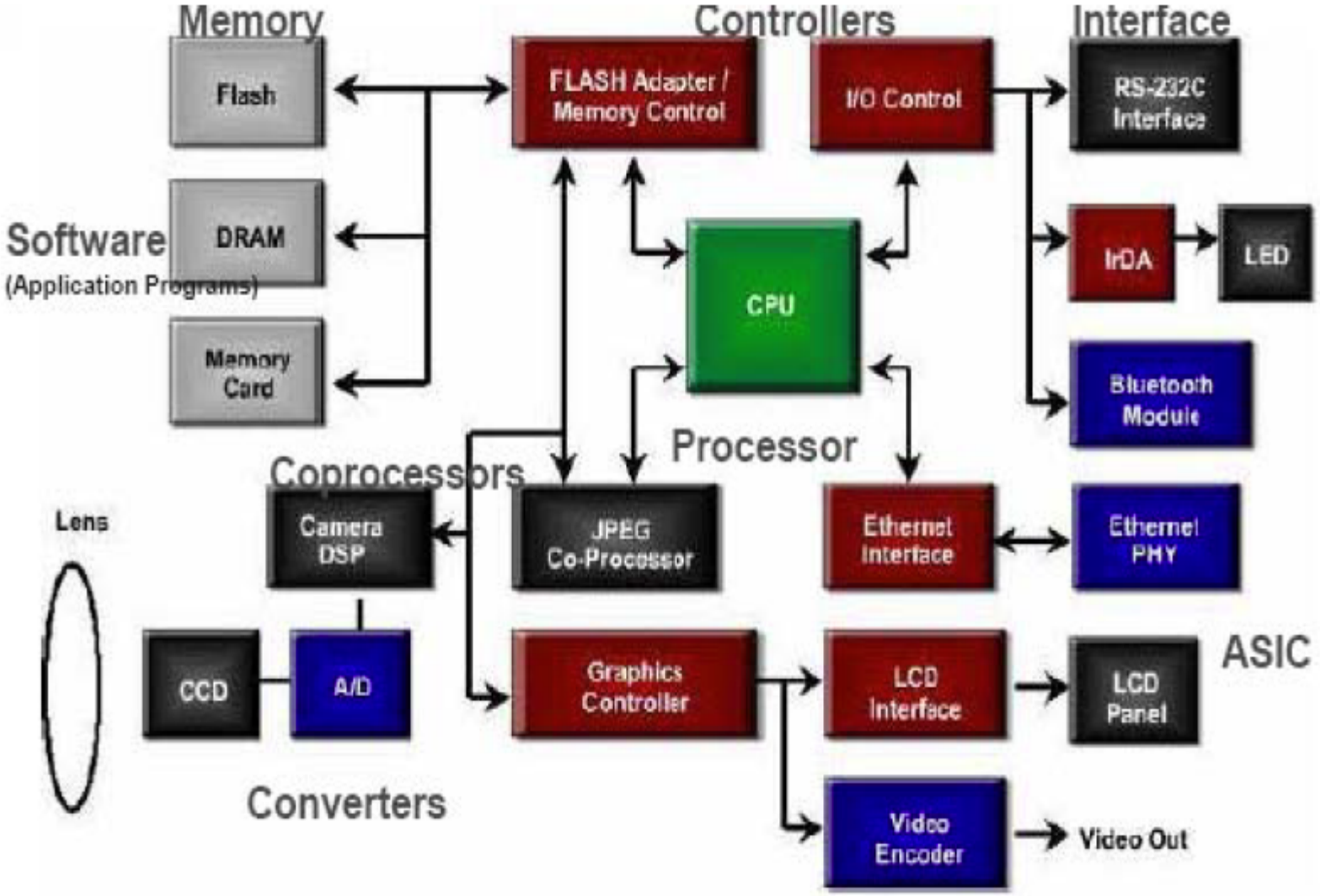
- Platform includes hardware, supporting software.
- Two stage process:
  - Design the platform.
  - Use the platform.
- Platform can be reused to host many different systems.



DESIGN PLATFORMS

- Different levels of integration:
  - PC + board.
  - Custom board with CPU + FPGA or ASIC.
  - Platform FPGA.
  - System-on-chip (SoC).

HARDWARE PLATFORM ARCHITECTURE



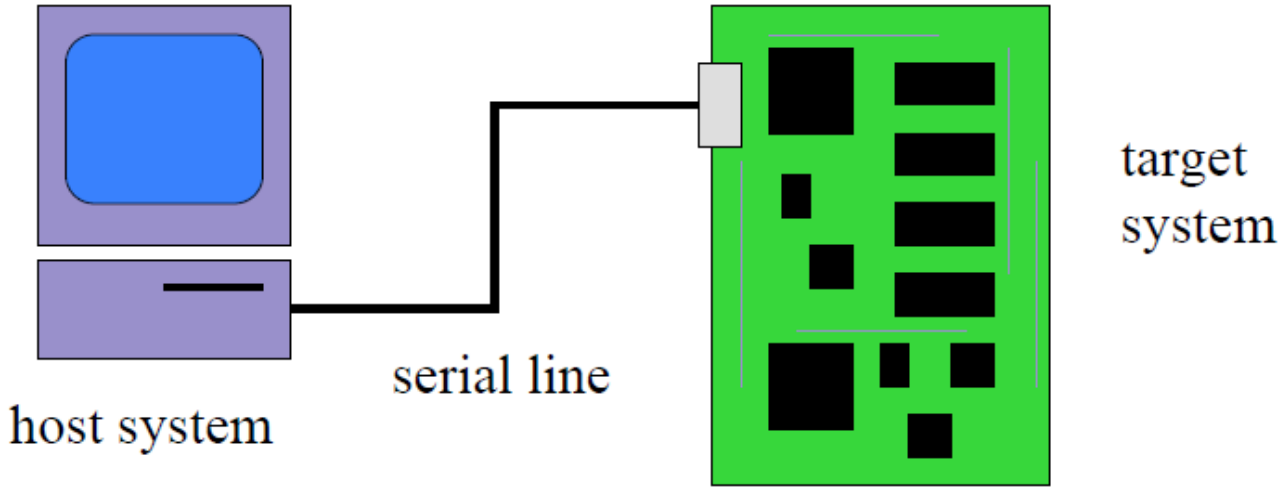
THE PC AS A PLATFORM

- Advantages:
  - Cheap and easy to get
  - Rich and familiar software environment

- Disadvantages:
  - Requires a lot of hardware resources
  - Not well-adapted to real-time

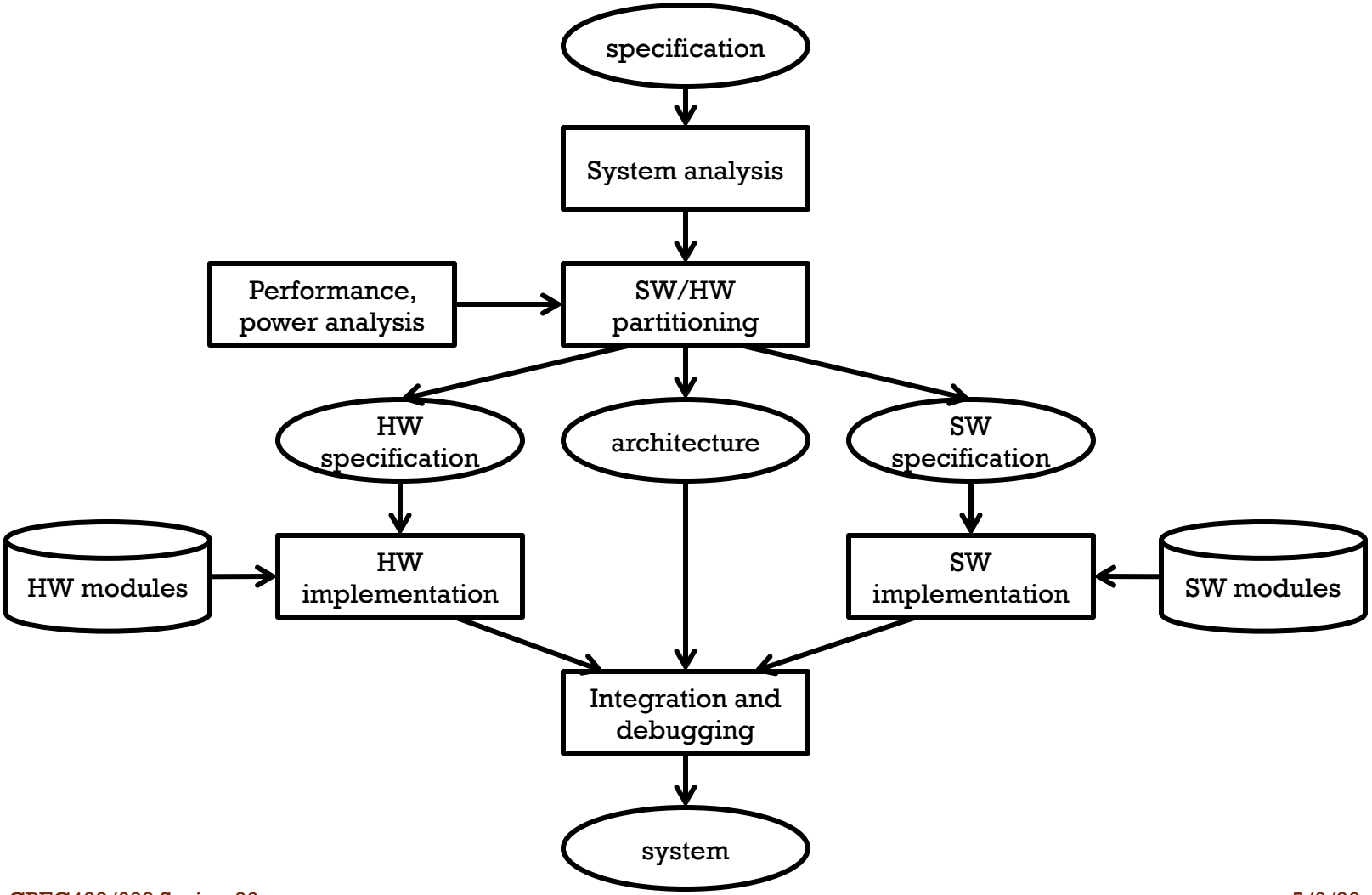
HOST / TARGET DESIGN

- Use a host system to prepare software for target system

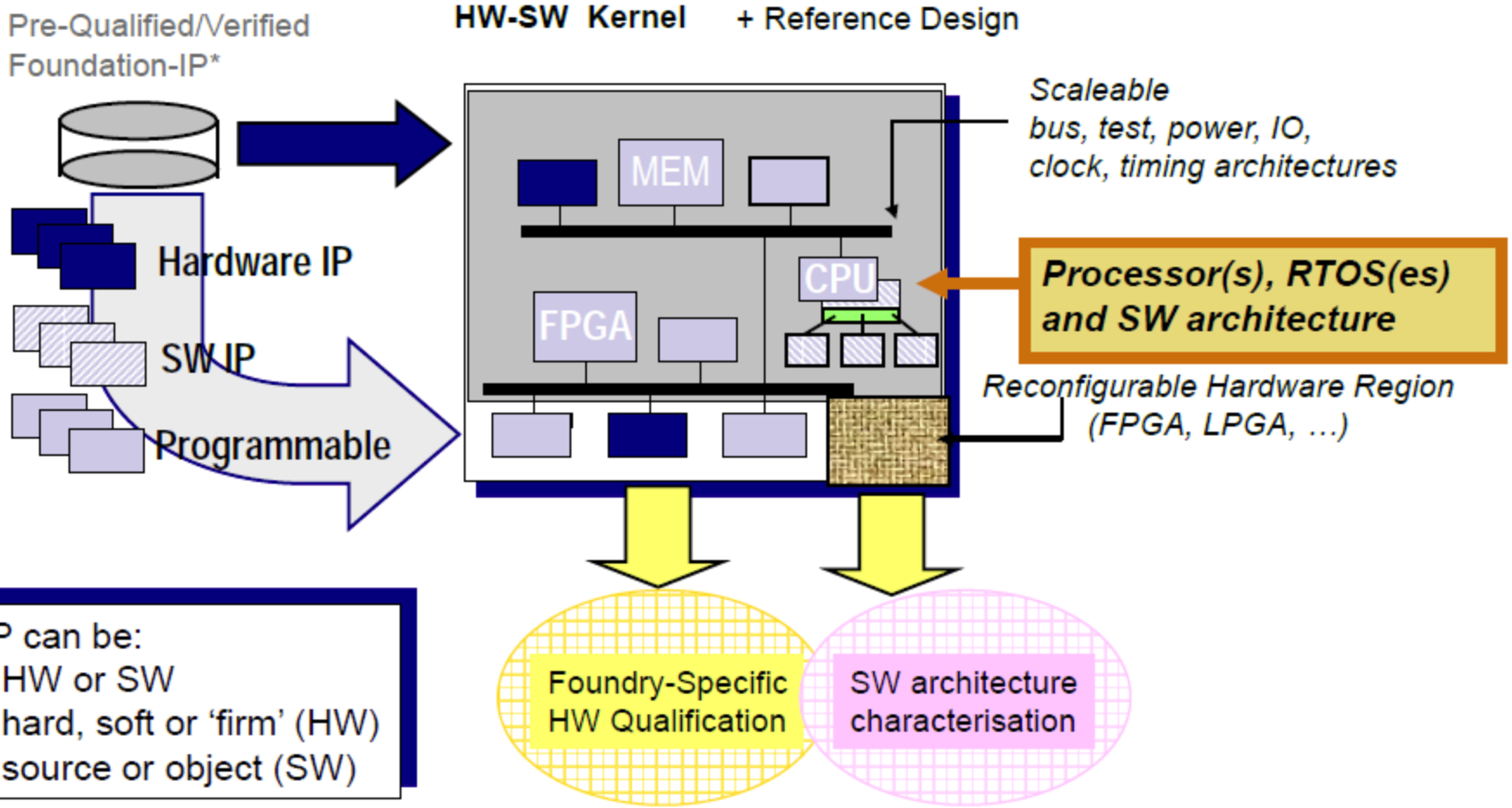


- Host-Based Tools
  - Cross compiler: Compiles code on host for target system
  - Cross debugger: Displays target state, allows target system to be controlled

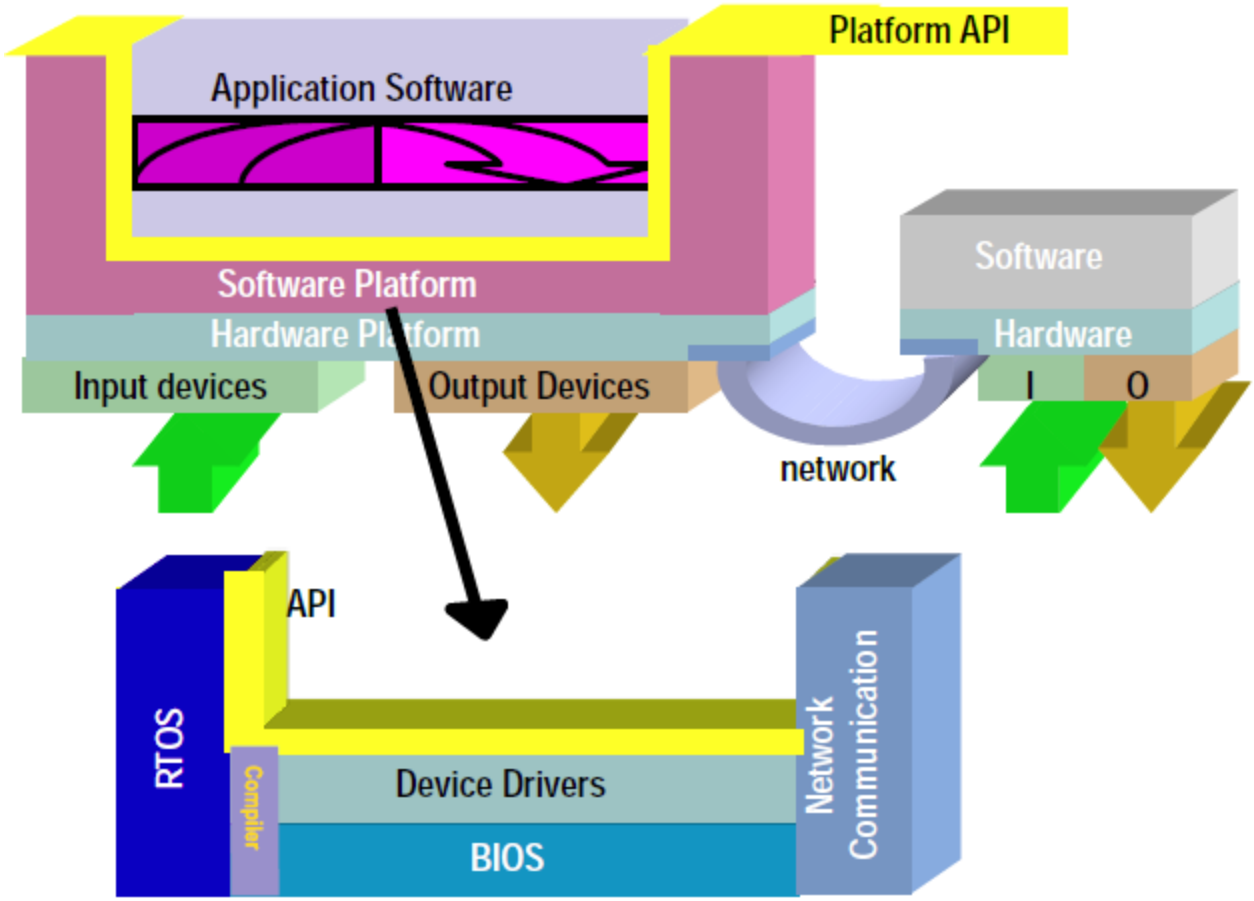
HARDWARE/SOFTWARE CO-DESIGN FLOW



HW-CENTRIC VIEW OF A PLATFORM



SW-CENTRIC VIEW OF PLATFORMS

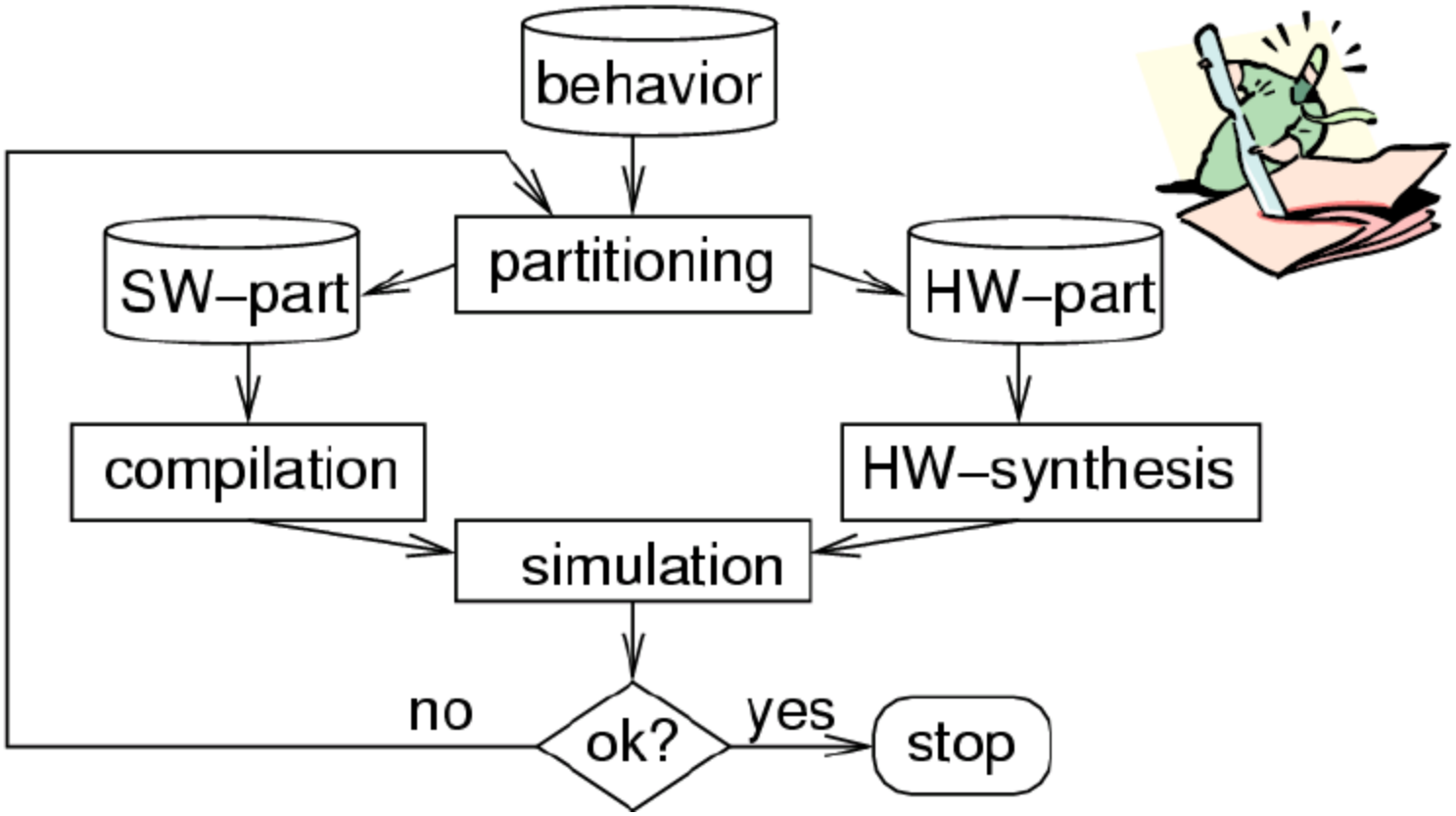


HW/SW CODESIGN ISSUES

- **Task level concurrency management**
  - Which tasks in the final system?
- **High level transformations**
  - Transformation outside the scope of traditional compilers
- **Hardware/software partitioning**
  - Which operation mapped to hardware, which to software?
- **Compilation**
  - Hardware-aware compilation
- **Scheduling**
  - Performed several times, with varying precision
- **Design space exploration**
  - Set of possible designs, not just one.

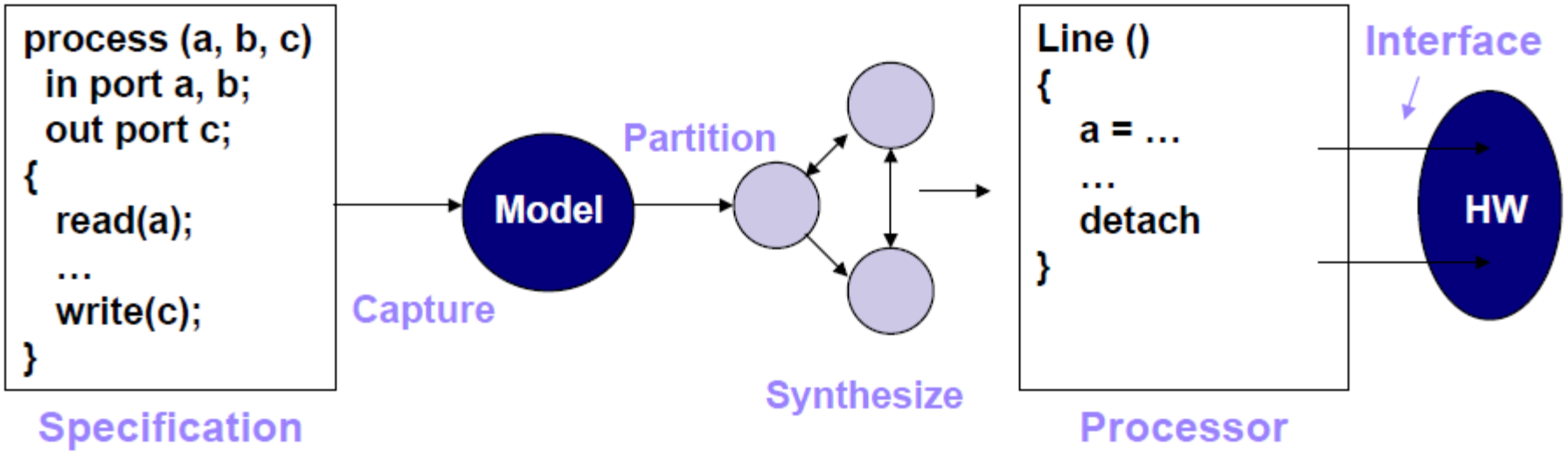
PARTITIONING

SOFTWARE OR HARDWARE?



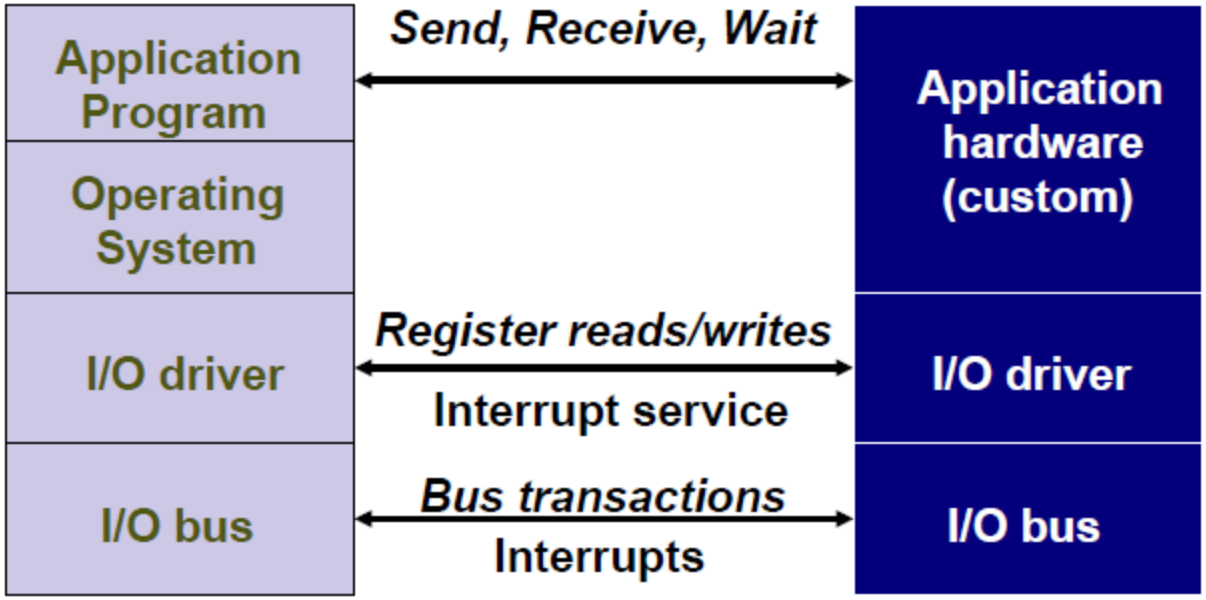
Decision based on hardware/ software partitioning

SYSTEM PARTITIONING



- Good partitioning mechanism:
  - Minimize communication across bus
  - Allows parallelism -> both HW & CPU operating concurrently
  - Near peak processor utilization at all times

DETERMINING COMMUNICATION LEVEL



- Easier to program at application level
  - (send, receive, wait) but difficult to predict
- More difficult to specify at low level
  - Difficult to extract from program but timing and resources easier to predict

PARTITIONING COSTS

- Software Resources
  - Performance and power consumption
  - Lines of code –development and testing cost
  - Cost of components
- Hardware Resources
  - Fixed number of gates, limited memory & I/O
  - Difficult to estimate timing for custom hardware
  - Recent design shift towards IP
    - Well-defined resource and timing characteristics

EXAMPLE: HW & SW FOUNDRIES

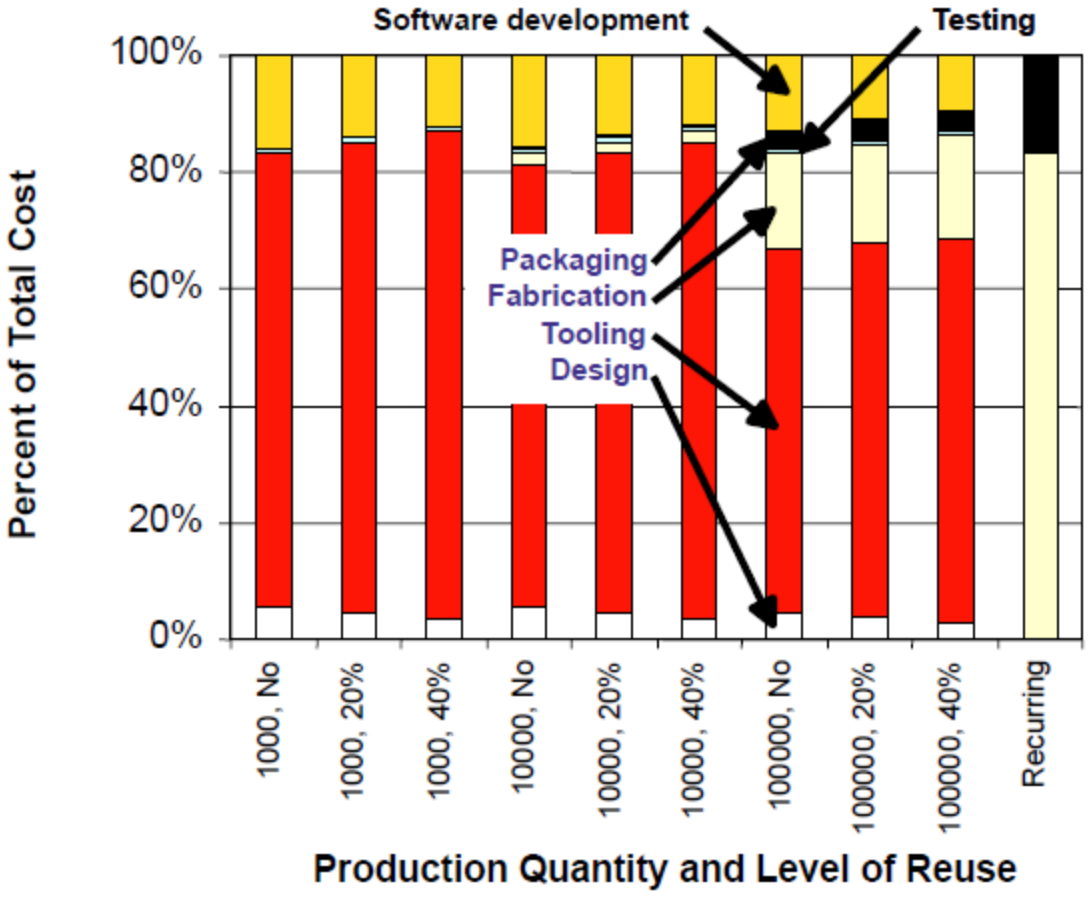
HW foundries

- HW1
  - LSI Logic ASIC Wafer Foundry Data
    - 0.18 μm feature size
    - 8 inch wafers
    - 6 layers
  - TSMC 018 Wafer Processing
- HW2
  - Samsung Semiconductor ASIC Wafer Foundry Data
    - 0.35 μm feature size
    - 6 inch wafers
    - 4 layers
  - TSMC 035 Wafer Processing

SW foundries

- SW1
  - Nominal to High development effort
- SW2
  - Low to Nominal development effort

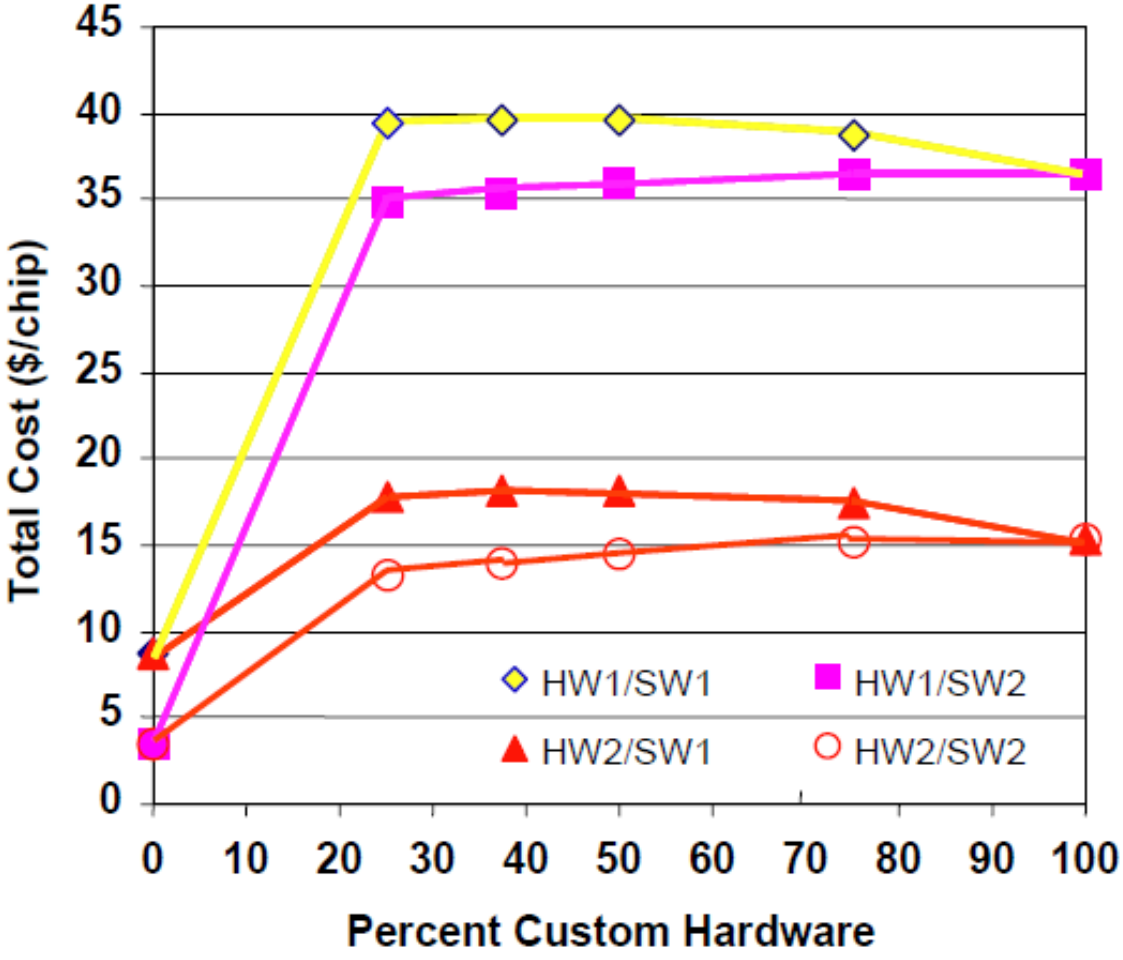
EXAMPLE: MIXED IMPLEMENTATION USING HW1 AND SW1



Reuse of:

- Gate-level IP
- Code

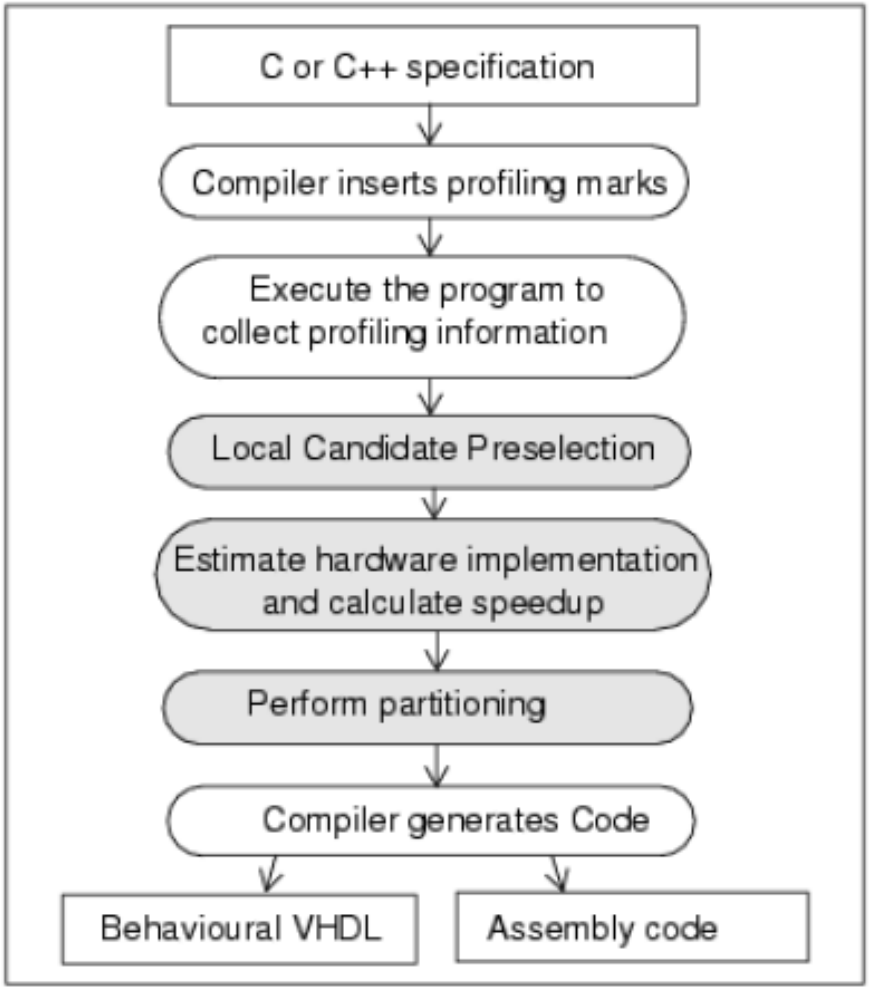
EXAMPLE: TOTAL COST PER CHIP



10,000 Units

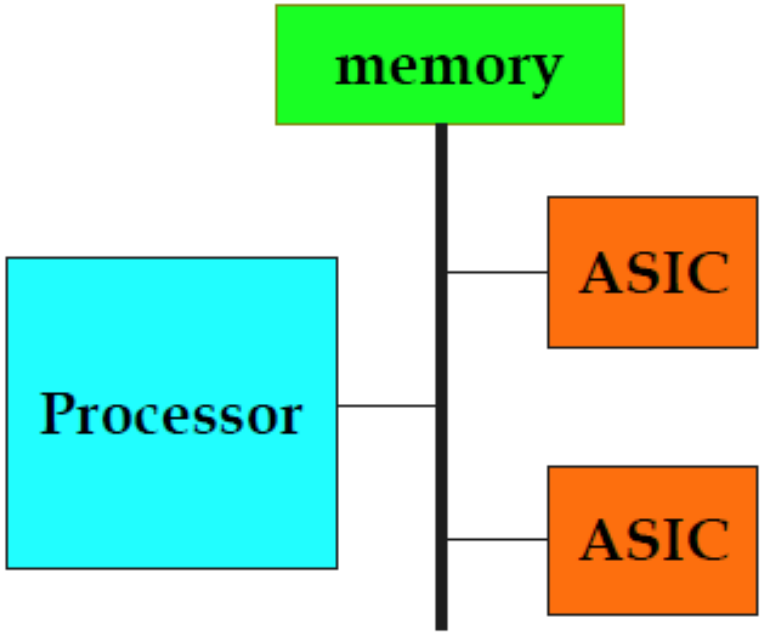
PARTITIONING ANALYSIS

- Result of compilation is synthesizable HDL and assembly code for the processor
- Compiler & profiler determine dependence and rough performance estimates



PARTITIONING STYLES

Simple architectural model: CPU + 1 or more ASICs on a bus



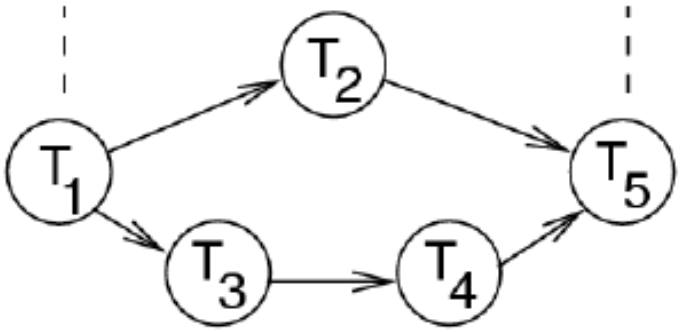
PARTITIONING STYLES

- HW first approach
  - start with all-ASIC solution which satisfies constraints
  - migrate functions to software to reduce cost
- SW first approach
  - start with all-software solution which does not satisfy constraints
  - migrate functions to hardware to meet constraints

SCHEDULING

SCHEDULING

- Goal: determine **when** and **where** a task is to be executed
- Inputs:
  - A task graph, node = task, edge = dependence



SCHEDULING

- Goal: determine **when** and **where** a task is to be executed
- Inputs:
  - A task graph, node = task, edge = dependence
  - Processor list, with different cost

- HW types H1, H2 and H3 with costs of 20, 25, and 30.
- Processors of type P.

SCHEDULING

- Goal: determine **when** and **where** a task is to be executed
- Inputs:
  - A task graph, node = task, edge = dependence
  - Processor list, with different cost
  - Task execution time on different processors

T	H1	H2	H3	P
1	20			100
2		20		100
3			12	10
4			12	10
5	20			100

SCHEDULING OUTPUTS

- For each task, determine
  - Its processor/HW assignment
  - Its starting time
- The overall hardware cost
  - The sum of costs of all needed units
- The overall schedule length
  - The finish time of the latest task
- The overall power consumption…

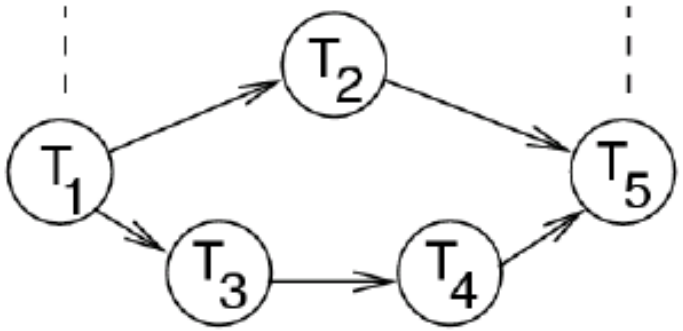
SCHEDULING CONSTRAINTS

- A task cannot start if any of its predecessor has not finished yet
- A processor/HW unit can execute only one task at any time

EXAMPLE 1:LOWEST-COST SCHEDULE

- To execute all 5 tasks we either
  - use P => cost = 50
  - use H1, H2, H3 => cost = 20+25+30 = 75
- Solution:
  - T1->P, start at time 0
  - T2->P, start at time 100
  - T3->P, start at time 200
  - T4->P, start at time 210
  - T5->P, start at time 220
  - Schedule length = 320

Cost	20	25	30	50
T	H1	H2	H3	P
1	20			100
2		20		100
3			12	10
4			12	10
5	20			100

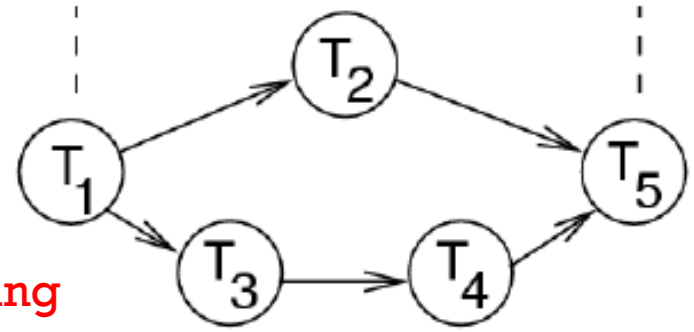


EXAMPLE 2:SHORTEST SCHEDULE

- Use the fastest HW for each task
  - T1, T5->H1
  - T2->H2
  - T3, T4->P
  - HW cost = 20+25+50 = 95
- Solution:
  - T1->H1, start at time 0
  - T2->H2, start at time 20
  - T3->P, start at time 20
  - T4->P, start at time 30
  - T5->H1, start at time 40
  - Schedule length = 60

Cost 20 25 30 50

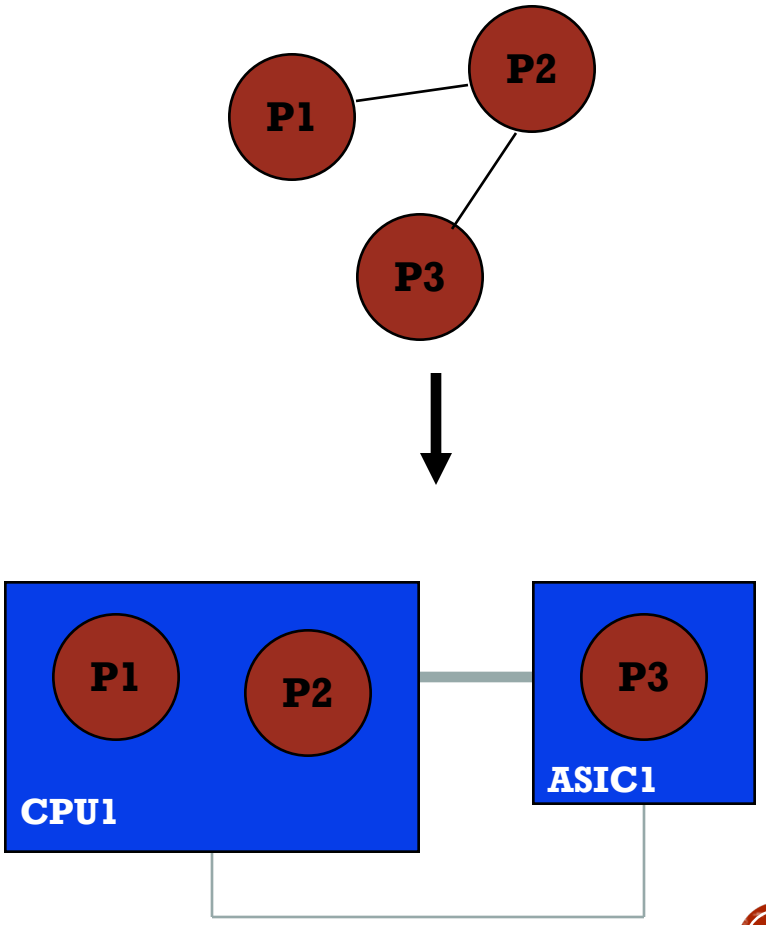
T	H1	H2	H3	P
1	20			100
2		20		100
3			12	10
4			12	10
5	20			100



The solution does not change even if assigning T2 on P takes a time of 10 instead of 100. **Why?**

COMMUNICATION COST

- Inter-processor communication cost is not zero!
- Must also schedule communication
- May impact schedule length significantly



CODESIGN MAPPING