Overview of Architectural-Level Synthesis
 (= High-Level Synthesis)
Key Synthesis/Optimization Steps: at 3 Levels

1. **Architectural Synthesis** (also, “High-Level Synthesis” [HLS])
   - **Starting point:** behavioral system specification
   - **Steps:** scheduling, resource allocation (sharing) and binding
   - **Outcome:** register-transfer level (RTL) optimized design
     for block-level datapath + FSM controller specification

2. **Logic Synthesis**
   - **Steps:**
     - **sequential synthesis:** FSM optimization
     - **combinational synthesis:** (i) 2-level logic minimization, (ii) multi-level logic optimization
     - **technology mapping:** optimal mapping of gates to VLSI “library” cells
   - **Outcome:** mapped gate-level circuit

3. **Physical Design**
   - **Steps:**
     - circuit partitioning, chip floorplanning, place-and-route (“P&R”)
     - ... + late timing correction/optimizations, etc.
   - **Outcome:** complete chip layout → ready for fabrication

---

Architectural Synthesis

High-Level Specification: Differential Equation Solver

(diff-eq) Custom Unit

```diff-eq
read (x, y, u, dx, a);
repeat {
xl = x + dx;
ul = u - (3 * x * u + dx) = (3 * y * dx);
yl = y + u * dx;
c = xl < a;
x = xl ; u = ul ; y = yl;
}
until (c);
write (y);
```

Architectural Synthesis

Target Micro-Architecture: Register-Transfer Level

Version #1

![Diagram 1.12]

FIGURE 1.12
Example of structural view at the architectural level.

p. 19, DM book

Architectural Synthesis

Target Micro-Architecture: Register-Transfer Level

Version #2

![Diagram 1.14]

FIGURE 1.14
Alternative example of structural view at the architectural level.

p. 22, DM book
More Detailed Target Micro-Architecture:

Figure 2. Typical architecture.


Architectural Synthesis: Deriving a CDFG Spec

High-Level Specification: Differential Equation Solver (diff-eq) Custom Unit

diffeq {
    read (x, y, u, dx, a);
    repeat {
        xu = x + dx;
        ul = u - (3 * x * u * dx) - (3 * y * dx);
        yl = y + u * dx;
        c = xu < a;
    }
    until (c);
    write (y);
}

xl = x + dx;
ul = u - (3 * x * u * dx) - (3 * y * dx);
yl = y + u * dx;
c = xl < a;

initial pseudo-code
extract pseudo-code of inner loop body

p. 121, DM book
Architectural Synthesis: Deriving a CDFG Spec

High-Level Specification: Differential Equation Solver
(diff-eq) Custom Unit

$pseudo-code$ of inner loop body

CDFG fragments

$p. 121, DM book$

Architectural Synthesis: Scheduling

Unscheduled Control-Dataflow Graph (CDFG): diff-eq

Final unscheduled CDFG specification
(constant operands omitted)

$p. 147, DM book$
**Architectural Synthesis: Scheduling**

**Scheduled CDFG: minimum-latency**

Resources:
- 4 MULT units
- 2 ALU's

Latency:
- 4 cycles

*P. 148, DM book*

---

**Architectural Synthesis: Scheduling**

**Scheduled CDFG: min-area = resource-constrained (RC)**

Resources:
- 1 MULT unit
- 1 ALU

Latency:
- 7 cycles

*P. 149, DM book*
Architectural Synthesis: Scheduling

Scheduled CDFG: min-area, under min-latency constraints

Resources:
- 2 MULT units
- 1 ALU

Latency: 4 cycles

p. 201, DM book

Architectural Synthesis: Resource Binding

p. 152, DM book