System Synthesis of Digital Systems

Petru Eles, Zebo Peng

Literature:

P. Eles, K. Kuchcinski and Z. Peng
"System Synthesis with VHDL"
Kluwer Academic Publisher, December 1997.

Examination:

- Term paper
- Seminar presentation

Introduction

1. Digital System Synthesis
2. Lecture Topics
3. Specification Domains and the Abstraction Hierarchy
4. The Y-Chart
5. Synthesis Steps
6. High Level Synthesis

What's About?

☞ It's about (automatic) synthesis of digital systems.

☞ A digital system (for this course)

- Consists of programmable processors and dedicated hardware (application specific integrated circuits - ASICs).
- Performs a well-defined task.

☞ **In this course there will be more emphasis on the specification and synthesis of the hardware part (the ASIC).**

Why Is This an Issue?

- Complexity: new computer aided methodologies are needed in the context of increasing complexity (SoC).
- Verification: starting from a (formal) system specification which is validated, and performing well defined design steps (transformations) with verifiable outputs.
- Time to market: only efficient tools and reuse can bring design productivity up to the expected level.

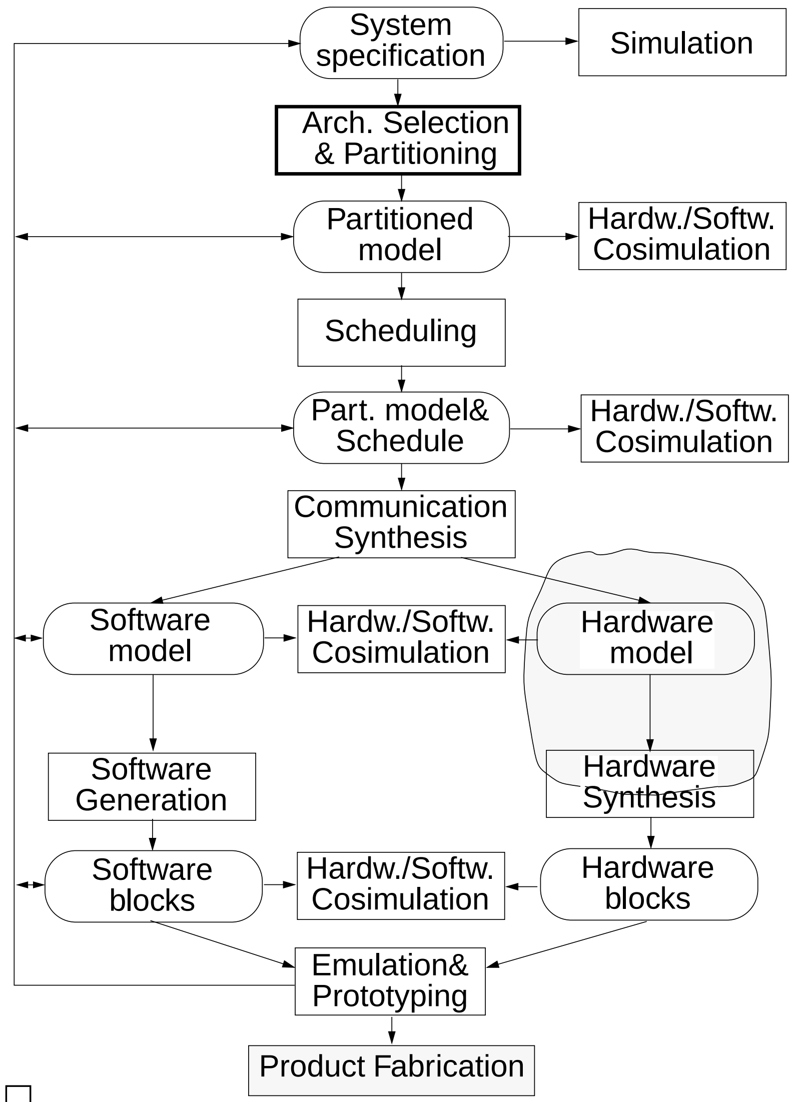
System Synthesis

Input: an implementation independent specification of the system; this includes: functionality and constraints.

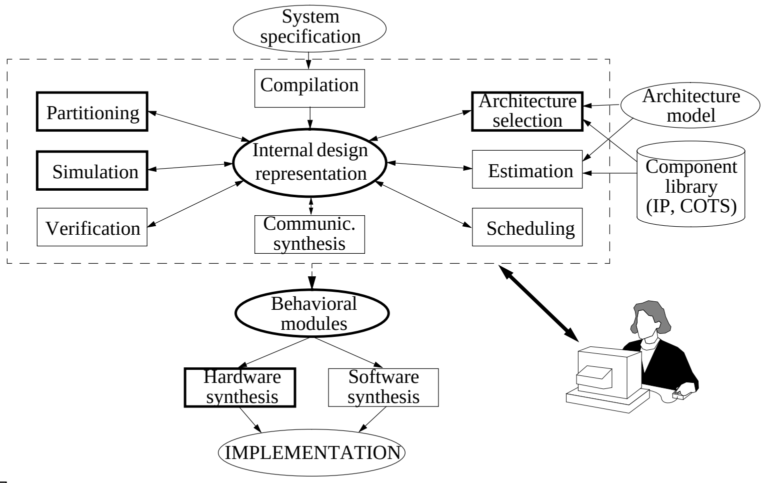
The synthesis tasks:

- To select the architecture;
- To partition functionality over the components of the architecture;
- To schedule activities
- To generate behavioral modules corresponding to the hardware and software domain of the implementation, including interface modules.
- The behavioral modules resulted from the previous steps are further synthesised into the actual hardware and/or software implementation.

System Synthesis (cont'd)



System Synthesis (cont'd)



Lecture Topics and Schedule

1. Introduction and Course Overview.
   System Synthesis and High-Level Synthesis.
   - Thursday March 23, 10-12.
2. VHDL - Basics and Simulation Mechanism.
   - Thursday April 6, 10-12.
3. High-Level Synthesis.
   - Thursday April 13, 10-12.
4. Basics of Transformational Approach.
   - Thursday April 20, 10-12.
5. Optimization Heuristics for Synthesis.
   - Thursday April 27, 10-12.
6. System-Level Synthesis and Hardware-Software Partitioning - I.
   - Thursday May 11, 10-12.
7. System-Level Synthesis and Hardware-Software Partitioning - II.
   - Thursday May 18, 10-12.
8. Synthesis of Advanced Features.
   - Wednesday May 31, 10-12.
9. High-Level Synthesis for Testability.
   - Thursday June 8, 10-12.
10. Presentation of Term Papers.
   - Thursday June 15, 10-??.

Digital Systems - Specification Domains

Functional Domain: emphasis is on behavior (input - output functionality), without any reference to the particular way in which this behavior is implemented.

Structural Domain: the specification is in terms of hierarchy of interconnected functional components.

Physical/Geometrical Domain: the specification is in terms of physical placement in space and physical characteristics without any elements to functionality.

Digital Systems - The Abstraction Hierarchy

System level:

The specification is given as a set of subsystems (modules/processes) which are loosely interacting (e.g. by exchanging messages).

The basic structural elements are processors, communication channels, ASICs, memories.

Algorithmic level (behavioral level):

The specification is given as an algorithm describing the functionality.

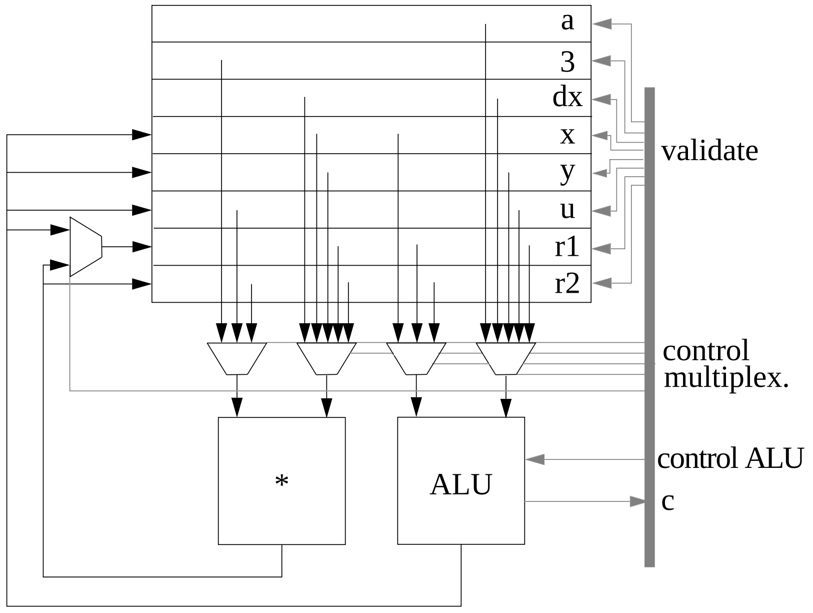
```
repeat
    xl = x + dx;
    ul = u - (3 * x * u * dx) - (3 * y * dx);
    yl = y + u * dx;
    c = xl < a;
    x = xl; u = ul; y = yl;
until (c);
```

The basic structural elements are controller and net-list.

The Abstraction Hierarchy (cont'd)

Register-transfer level:

Operations described as transfer of values between registers and functional units.



The basic structural elements are registers, ALUs, multiplexers and controller.

The Abstraction Hierarchy (cont'd)

Logic level:

Operations are described as boolean equations.

The basic structural elements are gates and their interconnections.

Circuit level:

Differential equations define relations between signal voltage, current response, etc.

The system is described in terms of transistors, resistors, capacitors.

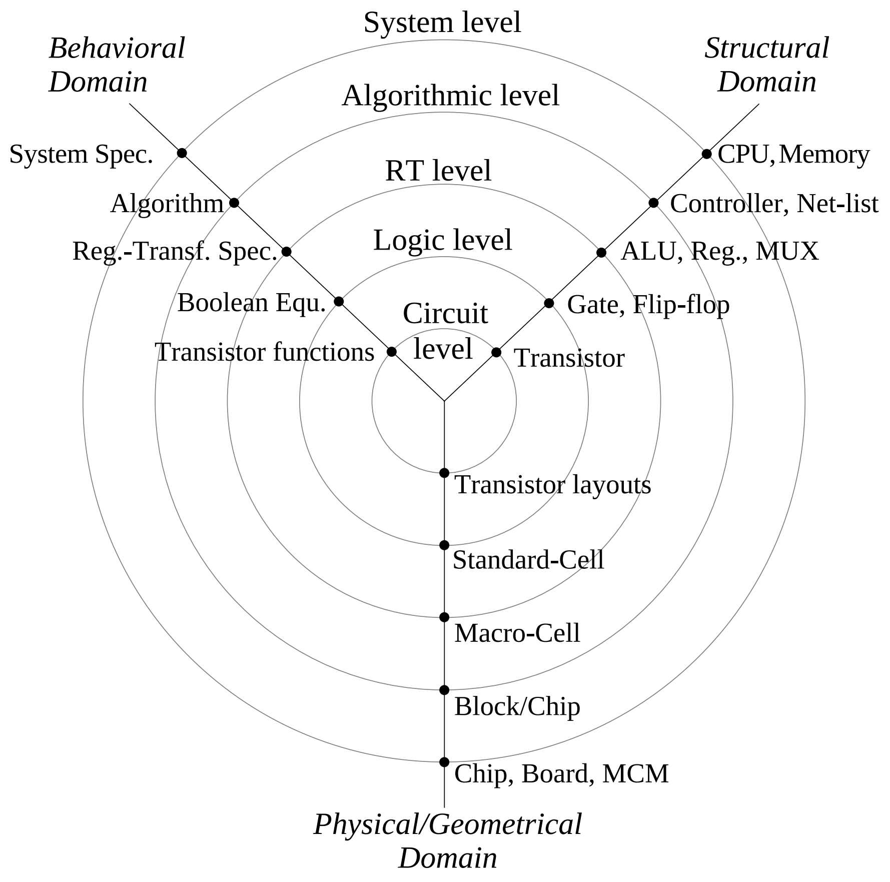
Y-Chart

☞ A representation proposed by Gajski, in order to capture specification domains, abstraction levels and their inter-relation.

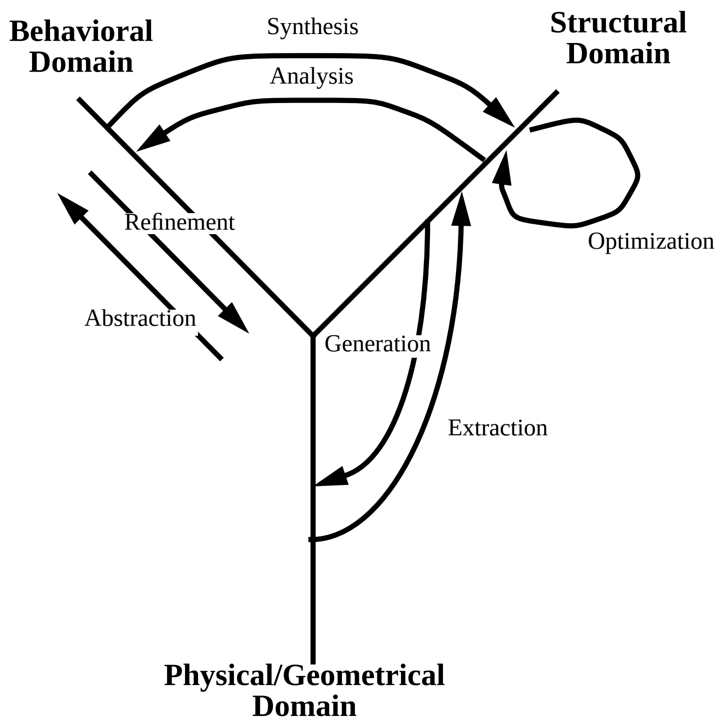
- Specification domains are represented as the three axes;
- In each domain, the specification can be at different abstraction levels. These levels are represented as points on the respective axes.

☞ The Y-chart also tries to capture the relation between different design activities (synthesis, analysis, refinement, etc.)

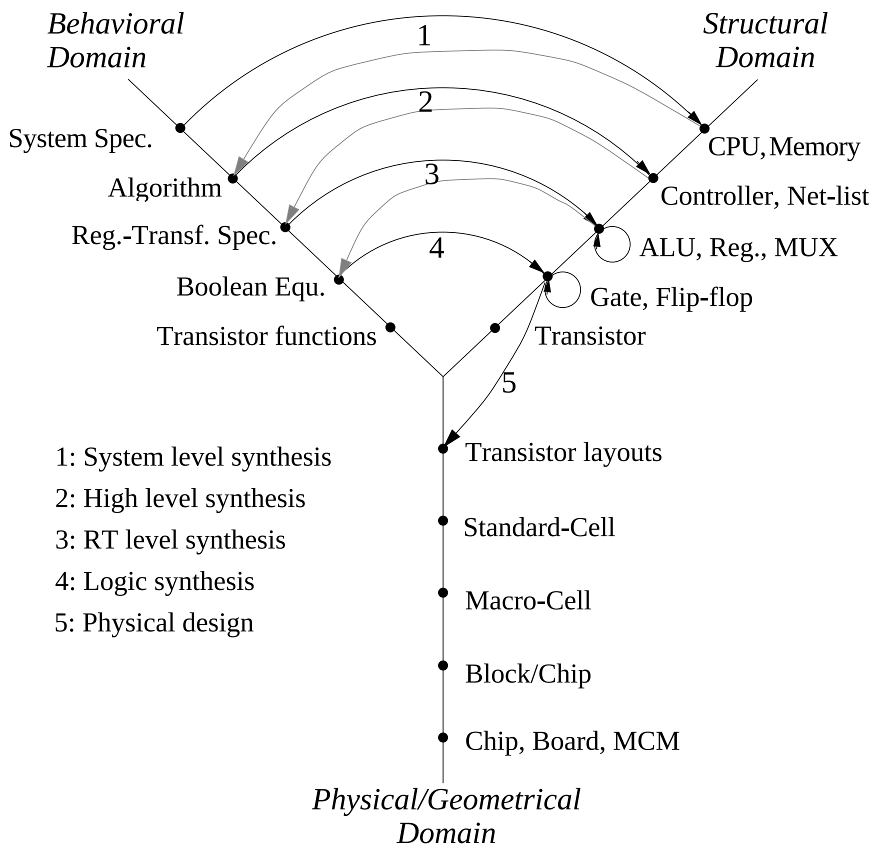
Y-Chart



Y-Chart and Design Activities



Top-Down Synthesis in the Y_Chart



1: System level synthesis
2: High level synthesis
3: RT level synthesis
4: Logic synthesis
5: Physical design

Synthesis Steps

Synthesis:

Transformation of a representation in the behavioral domain to a representation of the same design in the structural domain (at the same abstraction level).

The structural description which results after a synthesis step is formulated as an interconnection of abstract components.

Each such component is functionally specified at the following, lower abstraction level. These functional specifications are the input for the following synthesis step.

Synthesis Steps (cont'd)

System synthesis

Input: System level specification (interacting processes) + design constraints

Output: Behavioral elements to be synthesised to hardware and software + System architecture + Process schedule and mapping

High level (behavioral) synthesis

Input: Algorithmic description

Output: RT level description of the controller (FSM) + net-list (data path)

Synthesis Steps (cont'd)

RT level synthesis

Input: RT level description of the controller (FSM) + net-list

Output: Blocks of combinational and memory elements

Logic synthesis

Input: Blocks of combinational and memory elements (as boolean functions)

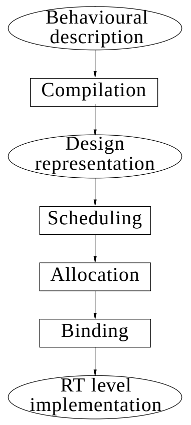
Output: gate-level net-list

Physical design

Input: gate-level netlist

Output: geometrical layout for a given technology

High Level Synthesis



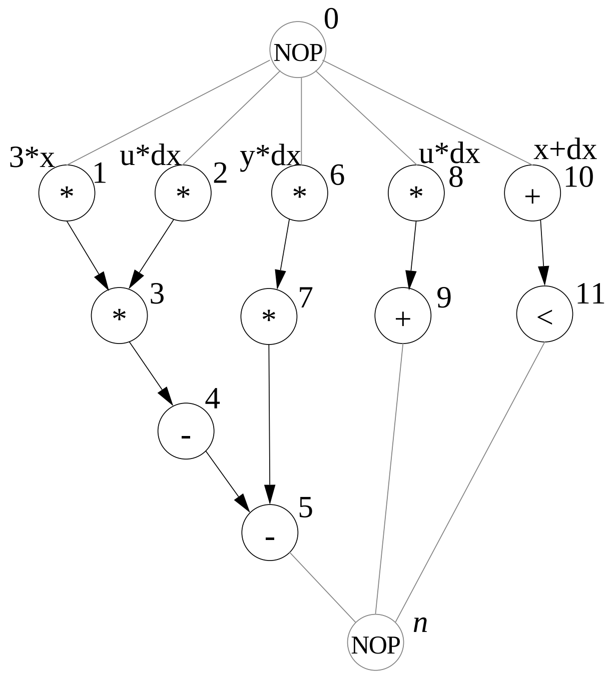
From Algorithm to Design Representation

$u = u - (3 * x * u * dx) - (3 * y * dx);$

$x = x + dx;$

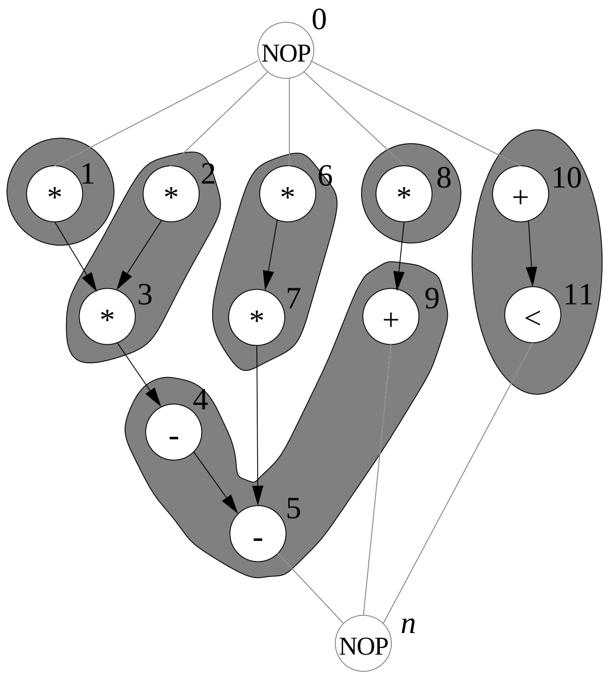
$y = y + u * dx;$

$c = x < a;$



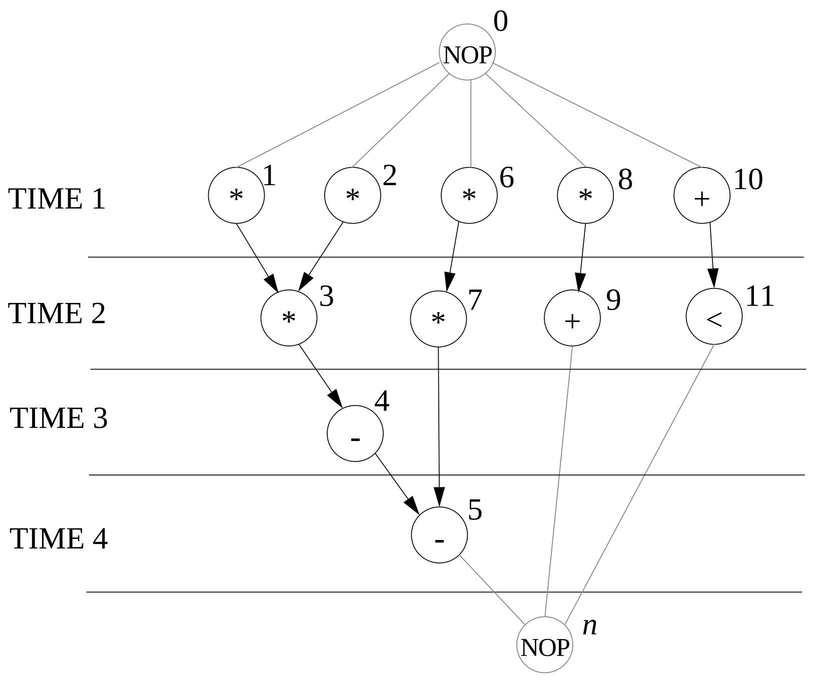
Allocation and Binding

- Allocate: 4 multipliers and two ALUs.
- Bind as follows:



Scheduling

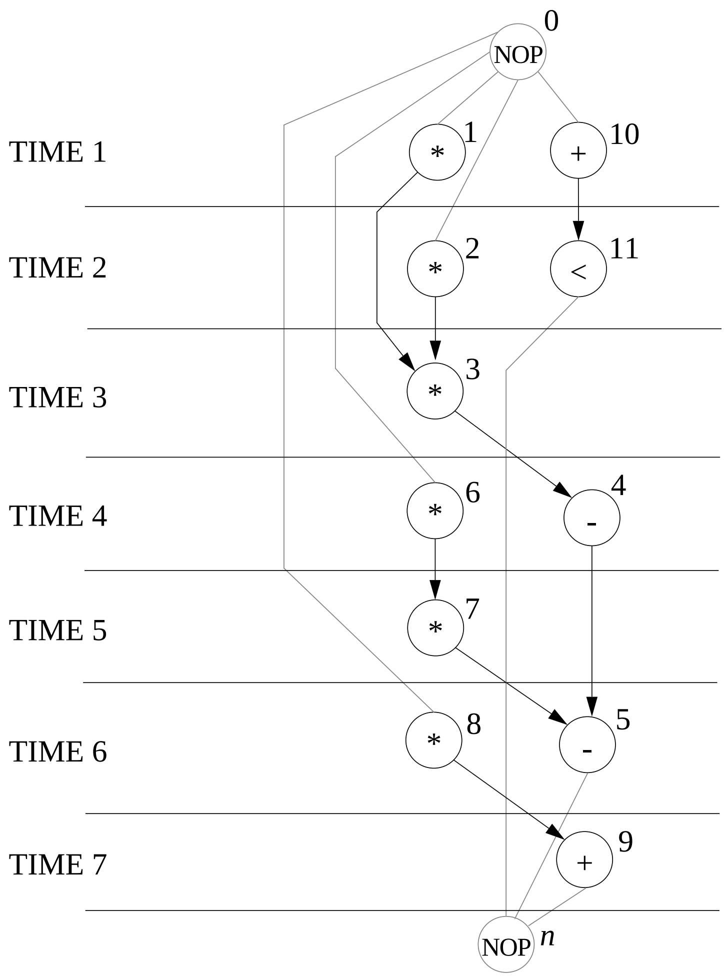
- Schedule the operations into clock cycles.



- We considered each operation to take one clock cycle.

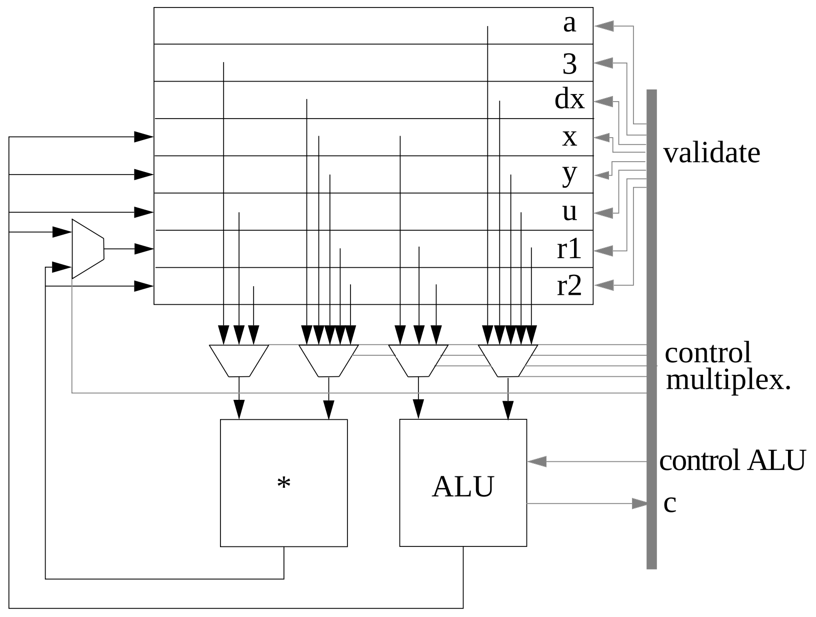
Scheduling (cont'd)

A schedule of the same graph, considering one multiplier and one ALU.



The Netlist

```
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);
```



The Controller

Suppose the following schedule (1 multiplier, 1 ALU):

v_1 : 3*x -> r2
v_{10} : x+dx -> r1,x

v_2 : u*dx -> r1
v_{11} : r1<a -> c

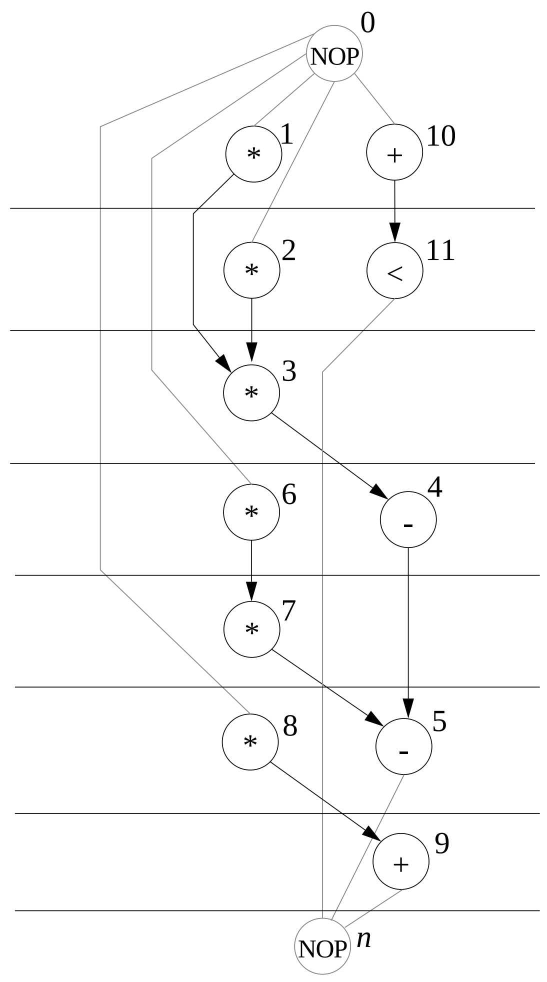
v_3 : r2*r1 -> r2

v_6 : 3*dx -> r2
v_4 : u-r2 -> r1

v_7 : r2*y -> r2

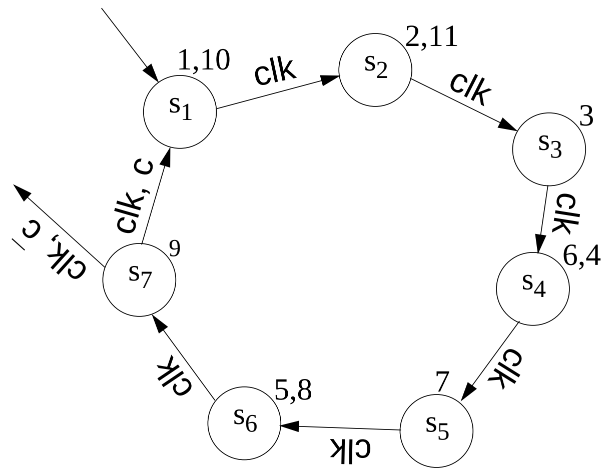
v_8 : u*dx -> r2
v_5 : r1-r2 -> u

v_9 : y+r2 -> y



The Controller (cont’d)

- One state for each clock cycle
- In each state the signals are generated which are needed in order to execute the operations scheduled for that cycle (see schedule).


s1
1,10
clk
s2
2,11
clk
s3
3
clk
s4
6,4
clk
s5
7
clk
s6
5,8
clk
s7
9
clk, c
clk, $\bar{c}$