High-Performance Embedded Systems-on-a-Chip

Sanjay Rajopadhye
Computer Science, Colorado State University

Lecture 3: Systolic Arrays & Systolic Synthesis

IRISA

Systolic Synthesis

Systolic Synthesis

With mathematical specification (SARE + reductions), do the following (not necessarily in order):

Systolic Synthesis

With mathematical specification (SARE + reductions), do the following (not necessarily in order):

1. Serialize reductions and Align inputs and outputs
2. Localize dependences
3. Schedule the SARE
4. Allocate the computation to processors
5. Transform the SARE
6. Generate the HDL

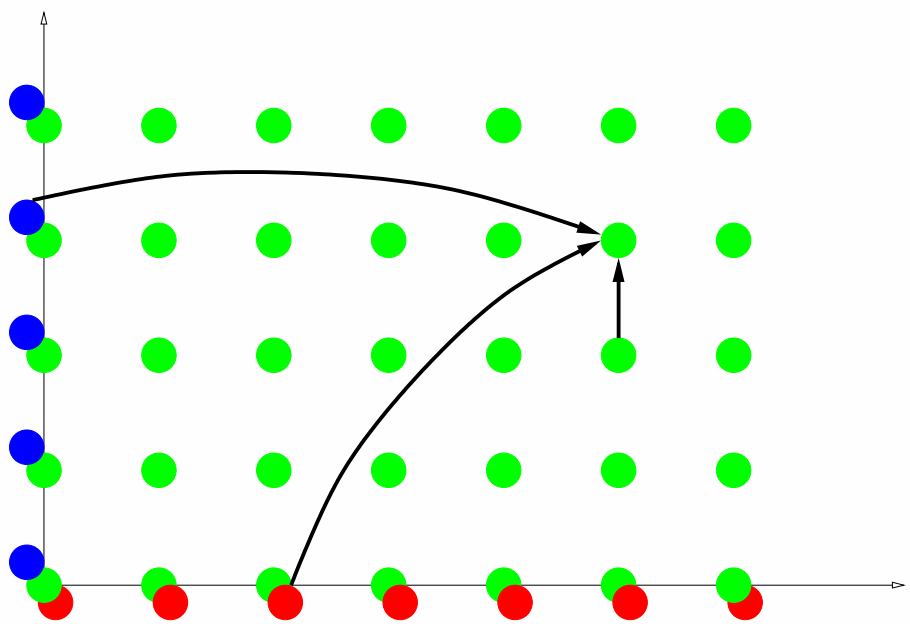
Example: Convolution

Initial specification:

$$y_i = \sum_{j=0}^{n-1} w_j * x_{i-j}$$

Serialization & Alignment

Replace (unbounded fan-in) $\sum$ by sequence of binary additions. Align input and output vars.

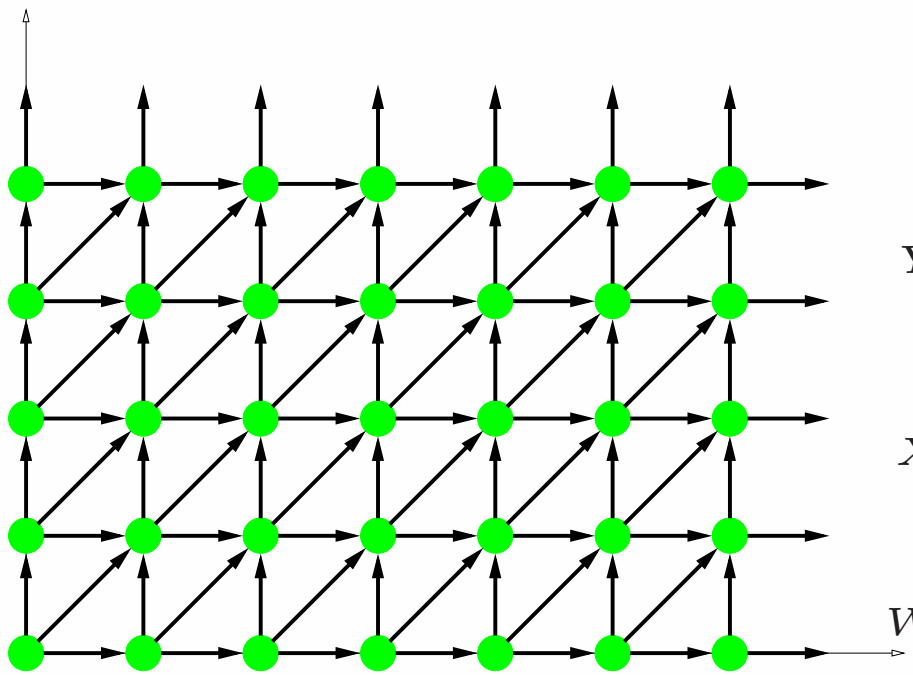


$$y_i = Y[i, n-1]$$

$$Y[i,j] = \begin{cases} j = 0 : w_j * x_{i-j} \\ j > 0 : Y[i, j-1] + \\ \qquad\qquad w_j * x_{i-j} \end{cases}$$

Localization/Uniformization

Remove unbounded fan-out (i.e., “long”) dependences.



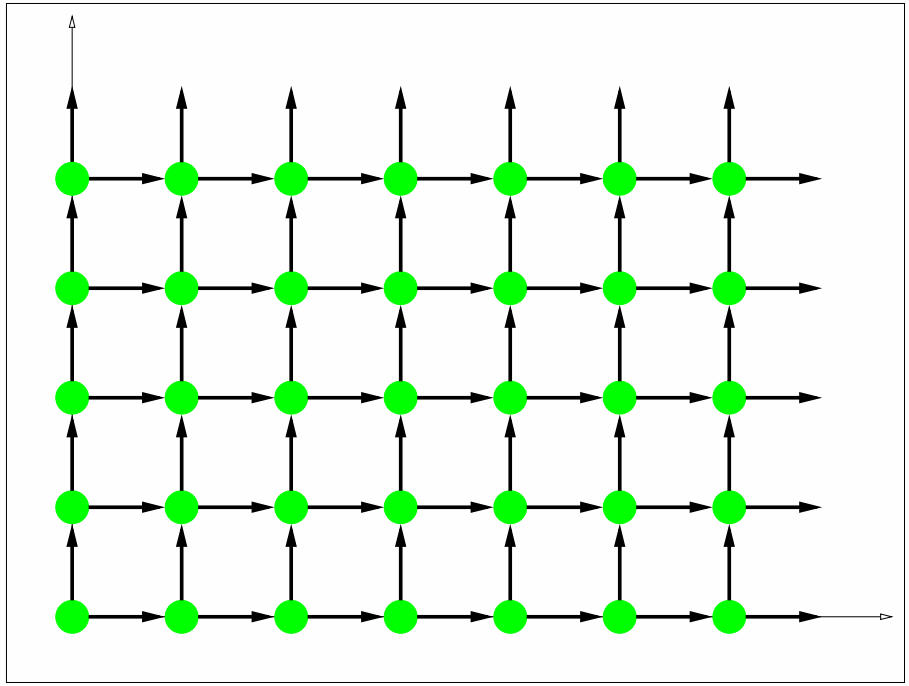
$$y_i = Y[i, n-1]$$

$$Y[i,j] = \begin{cases} j = 0 : & W[i,j] * X[i,j] \\ j > 0 : & Y[i,j-1] + \\ & \quad W[i,j] * X[i,j] \end{cases}$$

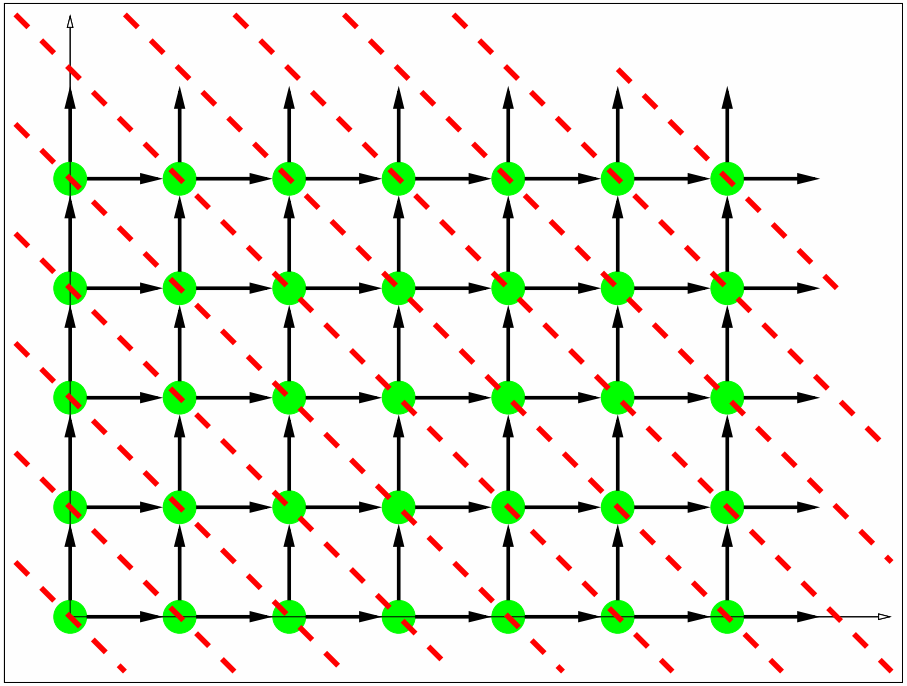
$$X[i,j] = \begin{cases} j = 0 : & x_i \\ j > 0 : & X[i-1, j-1] \end{cases}$$

$$W[i,j] = \begin{cases} i = 0 : & w_j \\ i > 0 : & W[i-1, j] \end{cases}$$

Scheduling & Allocation

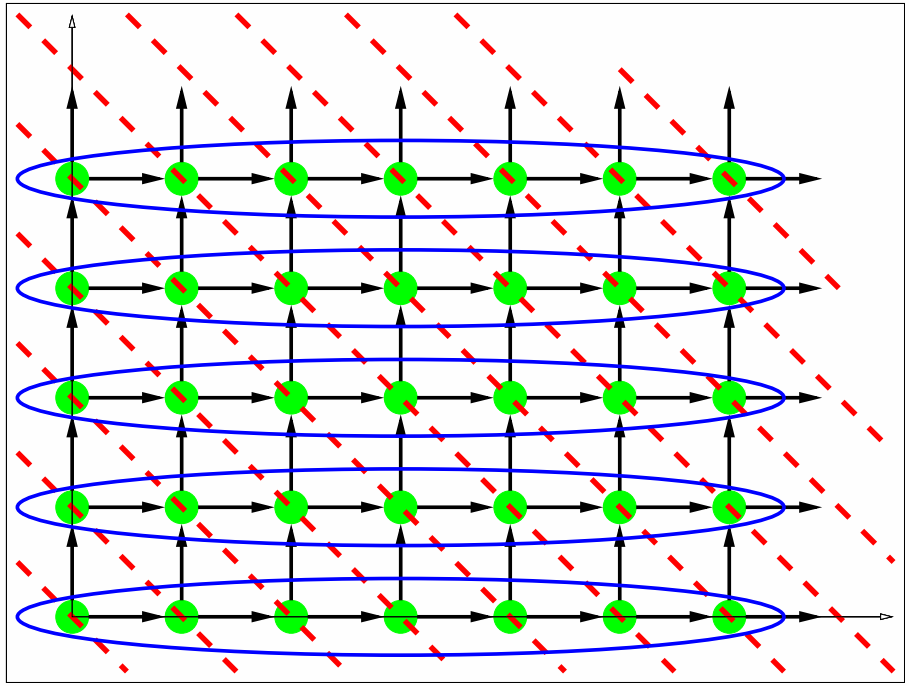


Scheduling & Allocation



A date: $t(i, j) = i + j$

Scheduling & Allocation

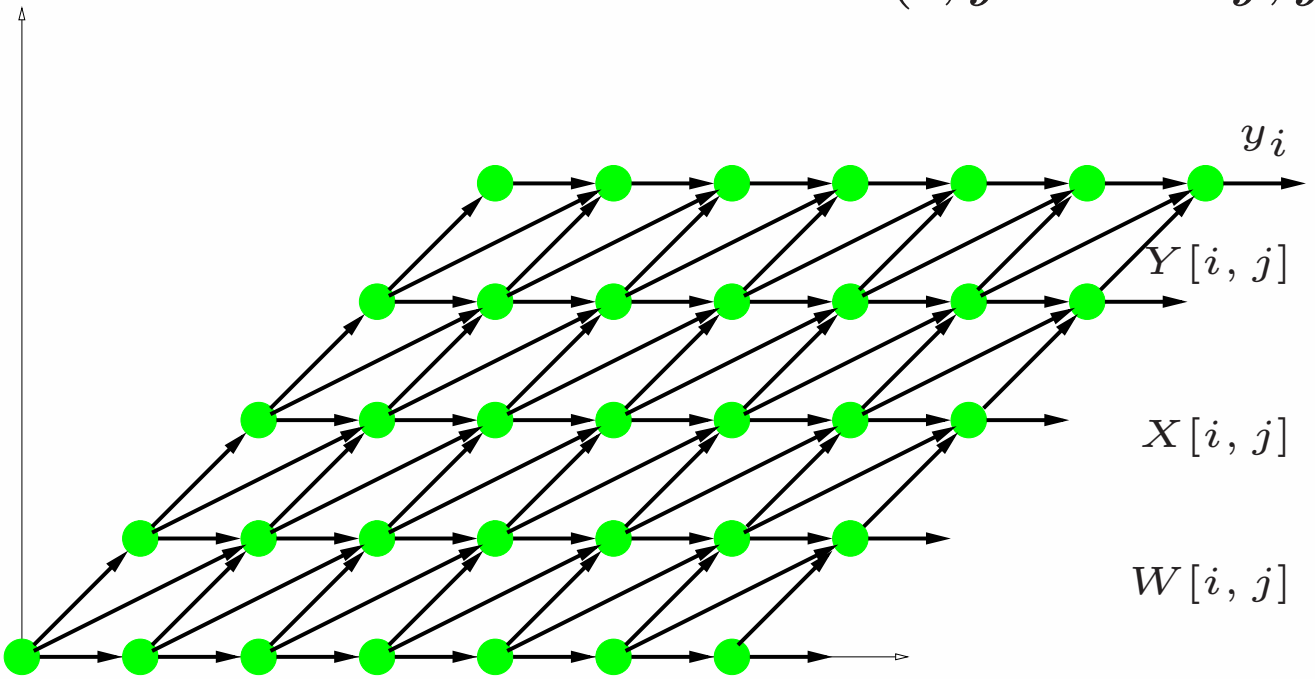


A date: $t(i,j) = i + j$

A place: $a(i,j) = j$

Geometric Transformation

$$\mathcal{T} = (i, j \rightarrow i + j, j)$$



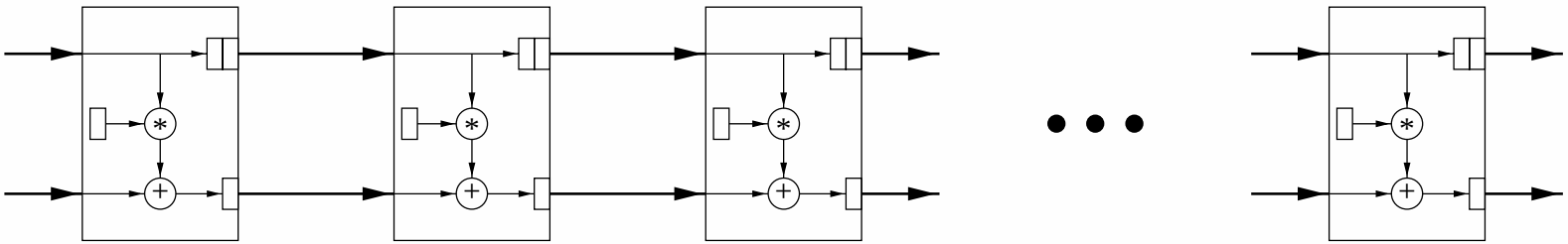
$$y_i = Y[i, n-1]$$

$$Y[i,j] = \begin{cases} j = 0 : W[i,j] * X[i,j] \\ j > 0 : Y[i-1, j-1] + \\ \qquad W[i,j] * X[i,j] \end{cases}$$

$$X[i,j] = \begin{cases} j = 0 : x_i \\ j > 0 : X[i-2, j-1] \end{cases}$$

$$W[i,j] = \begin{cases} i = j : w_j \\ i > j : W[i-1, j] \end{cases}$$

Generate HDL



IRISA

SREs as an HDL

Finite state machine:

$$\text{Next State} = \mathcal{N}(\text{Current state}, \text{Current input})$$

$$\text{Next Output} = \mathcal{O}(\text{Current state}, \text{Current input})$$

IRISA

SREs as an HDL

Finite state machine:

$$\begin{aligned} \text{Next State} &= \mathcal{N}(\text{Current state}, \text{Current input}) \\ \text{Next Output} &= \mathcal{O}(\text{Current state}, \text{Current input}) \end{aligned}$$

As an SRE:

$$\text{State}(t) = \begin{cases} t = 0 \ : \ \text{Init} \\ t > 0 \ : \ \mathcal{N}(\text{State}(t-1), \text{Input}(t-1)) \end{cases}$$

$$\text{Output}(t) = \left\{ t > 0 \ : \ \mathcal{N}(\text{State}(t-1), \text{Input}(t-1)) \right.$$

IRISA

SREs as a Systolic Array HDL

$$\text{State}(t,z) \quad = \quad \begin{cases} t = 0 \ : \ \text{Init}(z) \\ t > 0 \ : \ \mathcal{N}(\text{State}(t-1, z+\delta_z), \\ \qquad\qquad \text{Input}(t-1,z), \ldots) \end{cases}$$

$$\text{Output}(t,z) \quad = \quad \begin{cases} t = 0 \ : \ \text{Undefined} \\ t > 0 \ : \ \mathcal{N}(\text{State}(t-1, z+\delta'_z), \\ \qquad\qquad \text{Input}(t-1,z), \ldots) \end{cases}$$

Exercises

- Derive the convolution systolic array in Quinton-Robert. Hint: First *describe* the architecture and then try to reverse-engineer the design.
- Explore different allocation functions and schedules and derive alternate convolution array. Compare their respective performances in terms of area (number of processors), time, throughput and work.
- Describe a URE for the bubble sort algorithm (note: you cannot have data-dependent conditions). Derive two or more systolic arrays for it.
- Describe a URE for the band-matrix multiplication problem and derive systolic arrays for it.

More Exercises

Recall the rules for the change-of-basis transformations of SRE's. Consider the following SRE:

$$\begin{aligned} X[z] &= \{z \in D_X\} : g(\ldots, X[f_{xx}(z)], Y[f_{xy}(z)]) \\ Y[z] &= \{z \in D_Y\} : h(\ldots, X[f_{yx}(z)], Y[f_{yy}(z)]) \end{aligned}$$

and a bijective affine function, $\mathcal{T} : z \mapsto Tz + t$ where T is an integer unimodular matrix, and t is a vector. Show that when all the dependence functions are *uniform* (i.e., the SRE is actually a SURE) applying the *same* transformation, $\mathcal{T}$, to *both* the variables X and Y, the resulting system remains uniform. Determine the new dependence vectors. Can you think of a (slightly) more general transformation that retains this closure property?