EENG 426/CPSC 459/ENAS 876
Silicon Compilation

Process decomposition

Computer Systems Lab

http://csl.yale.edu/~rajit

Fall 2018

Process decomposition

Standard way to break up a large CHP program into smaller parts.

$P_1 \;\equiv\; \ldots;\; S;\; \ldots$

S: statement or group of actions

Break into:

$P_1 \;\equiv\; \ldots;\; C;\; \ldots$
$P_2 \;\equiv\; *[[\overline{C} \longrightarrow S; C]]$

C: synchronization channel

Process decomposition

Another option:

$P_2 \;\equiv\; *[[\overline{C} \longrightarrow S \bullet C]]$

for communication action S.

Process decomposition

Example:

$P \;\equiv\; \ldots\; x := x + 1;\; \ldots\; x := 0;\; \ldots\; y := x$

$P_1 \;\equiv\; \ldots\; I;\; \ldots\; J;\; \ldots\; K?y$

$$P_2 \;\equiv\; *[[\overline{I} \longrightarrow x := x + 1;\; I$$
$$[]\overline{J} \longrightarrow x := 0;\; J$$
$$[]\overline{K} \longrightarrow K!x$$
$$]]$$

Control/data separation

A common approach to designing chips: divide the design into

- Control: many unique components
- Datapath: high degree of replication of individual components, typically for computation/arithmetic operations

Separate compilation of control part and data manipulation part.

Control/data separation

Example:

$*[\ L?x;\ \ R!x\ ]$

x is a 32-bit variable.

Control part:

$*[\ L';\ \ R'\ ]$

Data part:

$*[\ L' \bullet L?x\ ]\ \|\ *[\ R' \bullet R!x\ ]$

Control/data separation

In general:

$$S\ \triangleright\ C_S \parallel D_S$$

- Replace all data communication with bare channels
- Add processes that only send/receive data

Control/data separation

Arithmetic operations:

- Transformations on the output of send operations
- Example: guard evaluation

Two different approaches to datapath:

- Robust to delays
- Standard combinational logic with matched delays