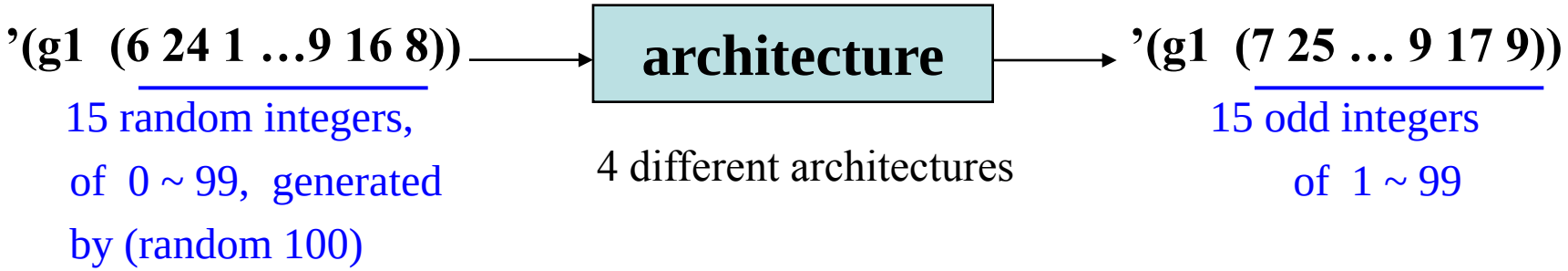
Final Project

Due : 5 pm, Jun. 19, 2020



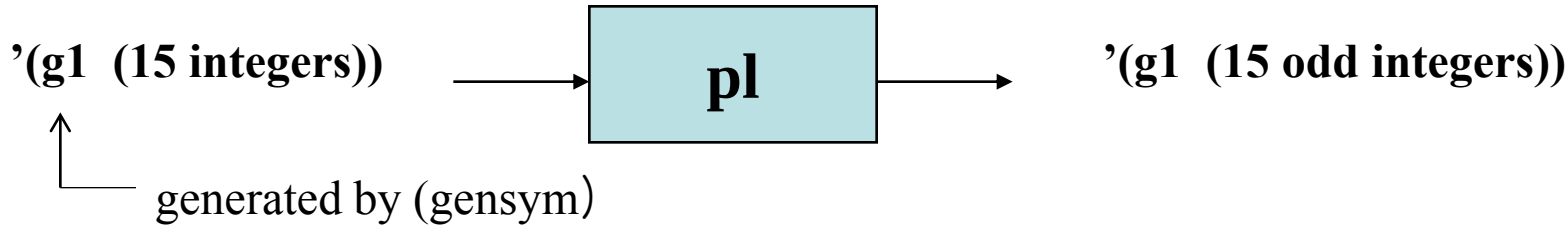
If a random integer is even then make the integer odd by adding 1 to the number.

The main objective of this project is to get the standard performance measures of the above 4 architectures (to be explained later). The standard performance measures are shown below.

Standard performance measures (or performances):

- average turn-around time
- throughput

pl model



Processing time : (truncate (unifrm 6 10 1))

returns the integers 6 ~ 9

Ex) '(g1 (90 8 7 32 18 21 56 87 4 7 2 7 4 1 66))

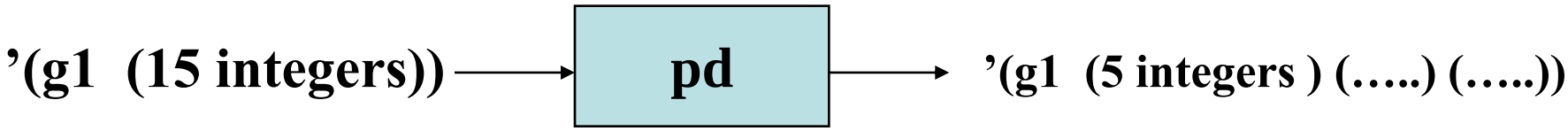
=> '(g1 (91 9 7 33 19 21 57 87 5 7 3 7 5 1 67))

psm model

Processing time : (truncate (unifrm 2 4 1))

Ex) '(g1 (90 8 7 32 18)) ➔ '(g1 (91 9 7 33 19))

pd (divider) model

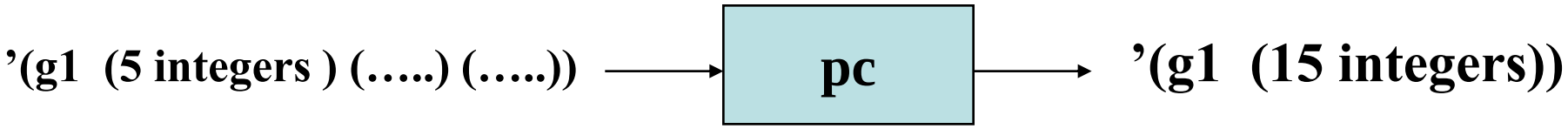


Processing time : (truncate (unifrm 2 4 1))

Ex) '(g1 (90 8 7 32 18 21 56 87 4 7 2 7 4 1 66))

=> '(g1 (90 8 7 32 18) (21 56 87 4 7) (2 7 4 1 66))

pc (compiler) model



Processing time : (truncate (unifrm 2 4 1))

Ex) '(g1 (91 9 7 33 18) (21 57 87 5 7) (3 7 5 1 67))

=> '(g1 (91 9 7 33 18 21 57 87 5 7 3 7 5 1 67))

- **mul-c, pip-c, dc-c model:**

  These models have 0 processing time.

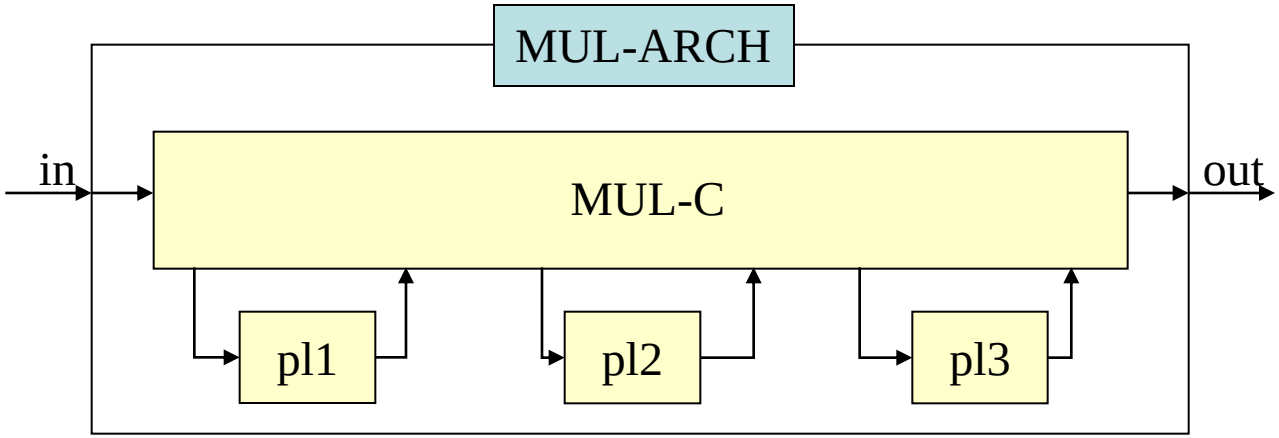
- **Code example of the random distribution function for sigma value assignment:**

```
. . .
(hold-in 'busy (truncate (unifrm 2 4 1)))
. . .
```

Architecture

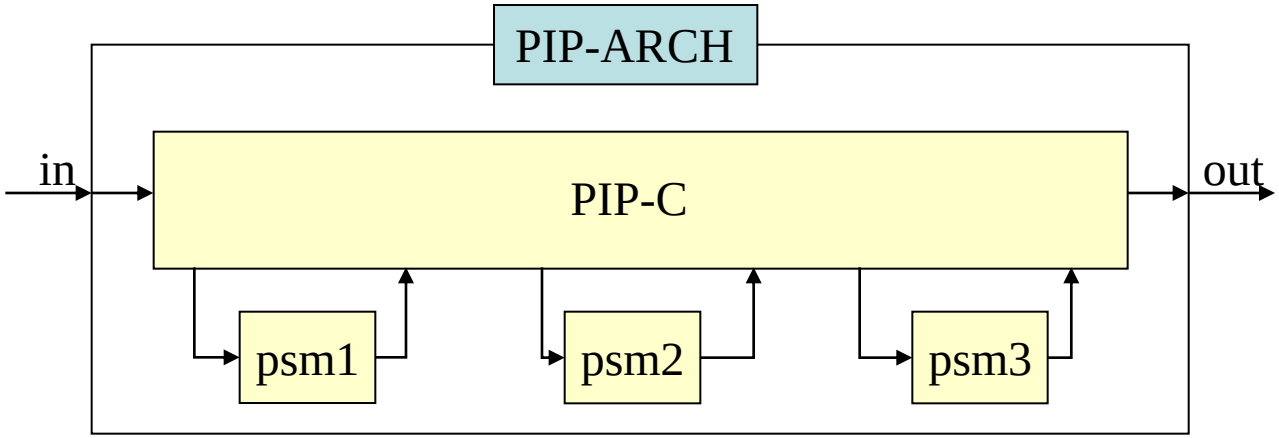
1) pl

2) multiserver



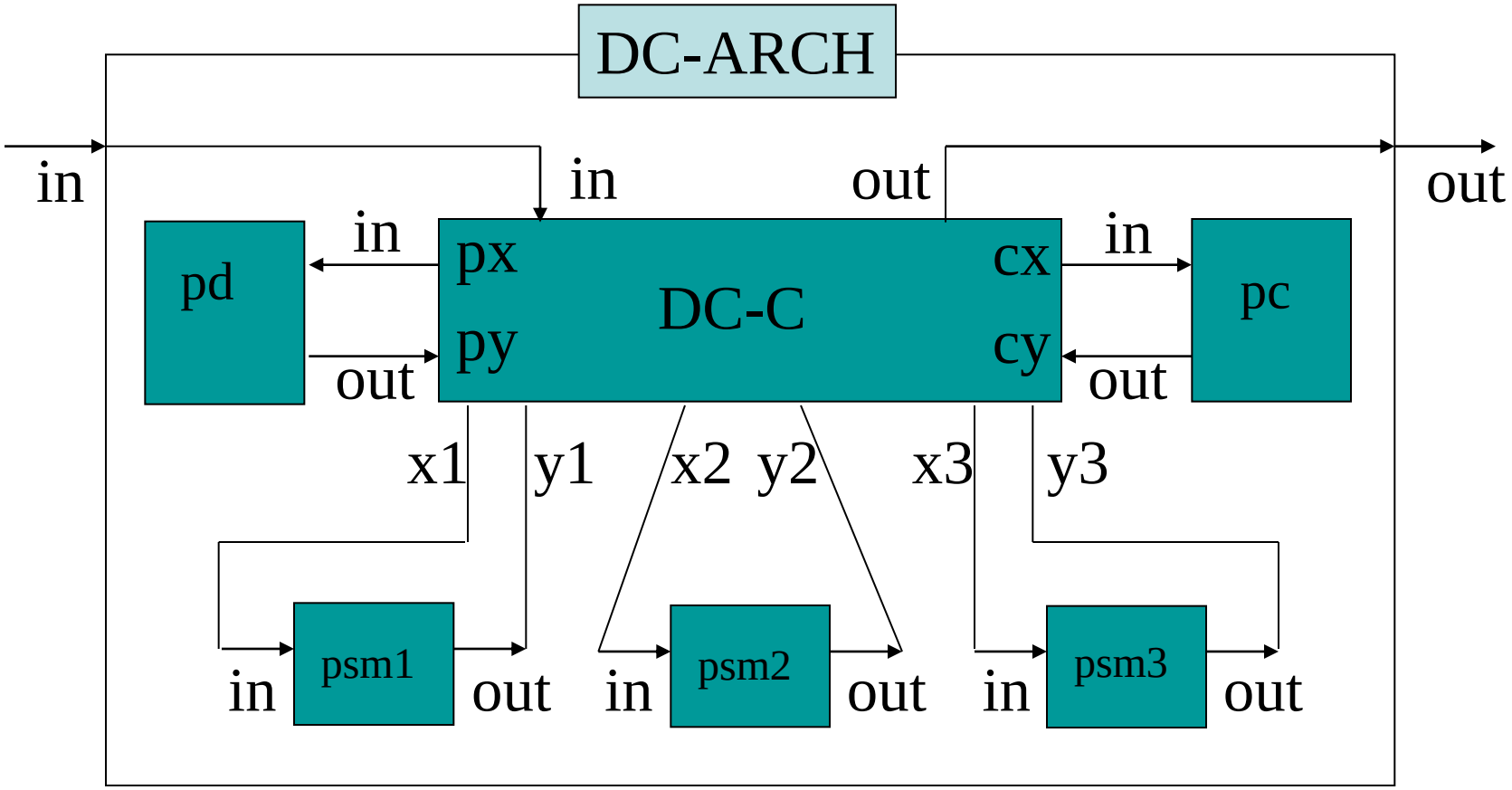
Architecture

3) pipeline (optional for extra points)



Architecture

4) divide&conquer



Simulation

- 5 samples : run simulation 5 times for each architecture with different seeds for the processing time generation (total 20 simulation runs or 15 without optional architecture)

- Ex) (unifrm 6 10 1)
  (unifrm 6 10 2)
  (unifrm 6 10 3)
  (unifrm 6 10 4)
  (unifrm 6 10 5)
  where 1,2,3,4,5 are seed values

Simulation

- genr
  - inter-arrival-time = 3
  - code for 15 random integers generation:

    (list (gensym) (list (random 100) … (random 100)))

    15 (random 100)

    where (random 100) returns an integer, 0 ~ 99

- transd
  - observation-interval = 100

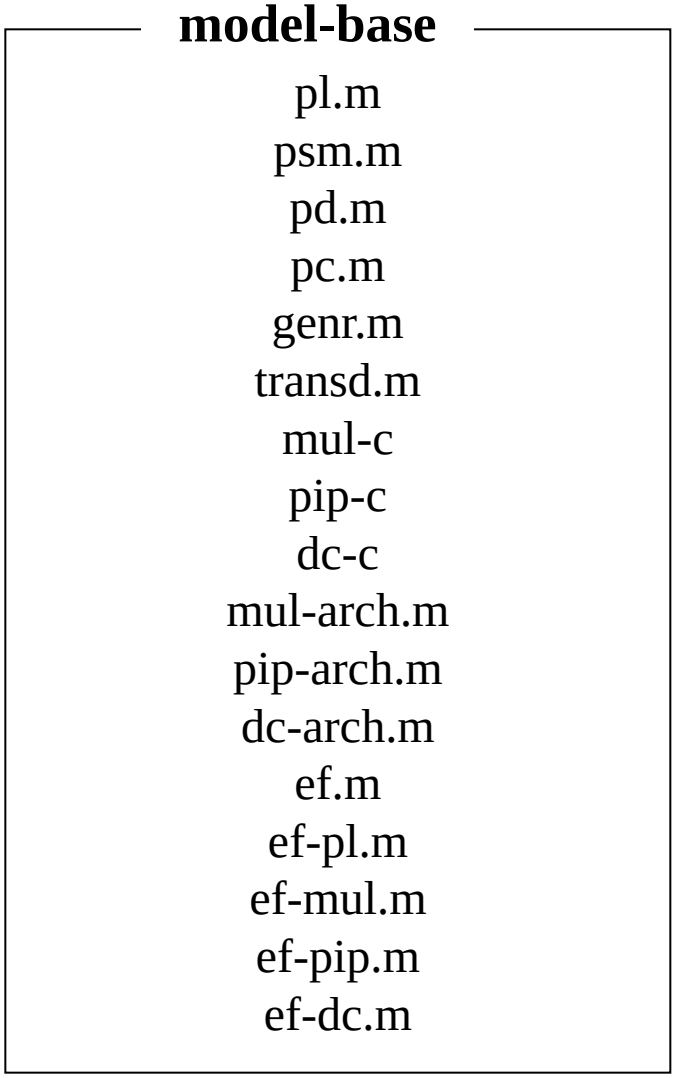
Simulation

- construct table as in p163 of “simulation” class note
  - theoretical case (as in table of p163) performances for the 4 different architectures
  - average of 5 performances (obtained from 5 simulation runs) for each of the 4 different architectures

    (3 architectures if pipeline architecture case is not included)
  - discussions on comparisons between

    the theoretical case and the experimental case performances

- C:\scheme\devs\final\mbase\



pip-c.m, pip-arch.m, ef-pip.m are optional for extra points

System Entity Structure

- Define a system entity structure (SES) that represents 4 different architectures (pl, mul, pip, dc). (or 3 architectures if pipeline is not included)
- Prune and transform for the 4 architectures (or 3 architectures) using the SES.
- Execute the simulation just once for each of the 4 architectures.
  (Just to find out the effectiveness of SES.)

- Submit the followings in a final project file (*.hwp or *.doc)

Orders within this file :

- contents (목차) of the this file
- table (performances of theoretical and simulation cases)
  and discussions on the performance differences between the theoretical case and the simulation case, and also within the theoretical case and the simulation case
- 3 diagrams for pl, psm, pd, pc atomic models
  (3 diagrams: model, timing, state transition diagrams)
- tests (verifications) of pl, psm, pd, pc atomic models
- 20 log files (simulation results of 20 runs, or 15 runs without pipeline),
  where is the log files are generated by transd model
  (not the file generated by start-log function)
- 17 model program files (14 models without pipeline)
  (pl.m, psm.m, pd.m, …, ef-dc.m)
- SES source codes (SES file)
- SES diagram and screen copy of prune and transform operations

How to submit the final project

- Create a **zip file** and **email the zip file** to **wonjin12@skku.edu**

  The zip file should be named as shown below:

  final-project_id_name.zip

  Ex) final-project_2020123456_HongGilDong.zip

- Above **zip file** should contain the following files:

  1) final project file (*.hwp or *.hwp) - shown in the previous page

     ↳ project file name is same as the zip file name

  2) The 17 model files (or 14 if pipeline is not included)

     (pl.m, psm.m, pd.m, …, ef-dc.m)

start-log (for coupled model test)

- Directions for Use

  [1] (start-log “file-name”)

  ⋮

  [5] (stop-log)

- Example

  [1] (start-log “ef-p-test.log”)

  [2] (load “mbase/ef-p.m”)

  [3] (initialize r c:ef-p)

  [4] (restart r)

  ⋮

  [5] (stop-log)

ef-p-test.log

```
clock time: 0
SUBP input x = IN JOB-8
P state s = (10 BUSY JOB-8 10)

GENR state s = (3 ACTIVE 3)

clock time: 3
P input x = IN JOB-9
P state s = (7 BUSY JOB-8 10)
TRANSD input x = ARIV JOB-9
TRANSD state s = (47 ACTIVE ((JOB-9 3) (JOB-8 0)) - 3 0)

GENR state s = (3 ACTIVE 3)

clock time: 6
P input x = IN JOB-10
P state s = (4 BUSY JOB-8 10)
TRANSD input x = ARIV JOB-10
TRANSD state s = (44 ACTIVE ((JOB-10 6) (JOB-9 3) (JOB-8 0)) - 6 0)

GENR state s = (3 ACTIVE 3)

clock time: 9
P input x = IN JOB-11
P state s = (1 BUSY JOB-8 10)
TRANSD input x = ARIV JOB-11
TRANSD state s = (41 ACTIVE ((JOB-11 9) (JOB-10 6) (JOB-9 3) (JOB-8 0)) - 9 0)
```