Petri Net Modeling, Scheduling and Real-Time Control of Cluster Tools in Semiconductor Manufacturing

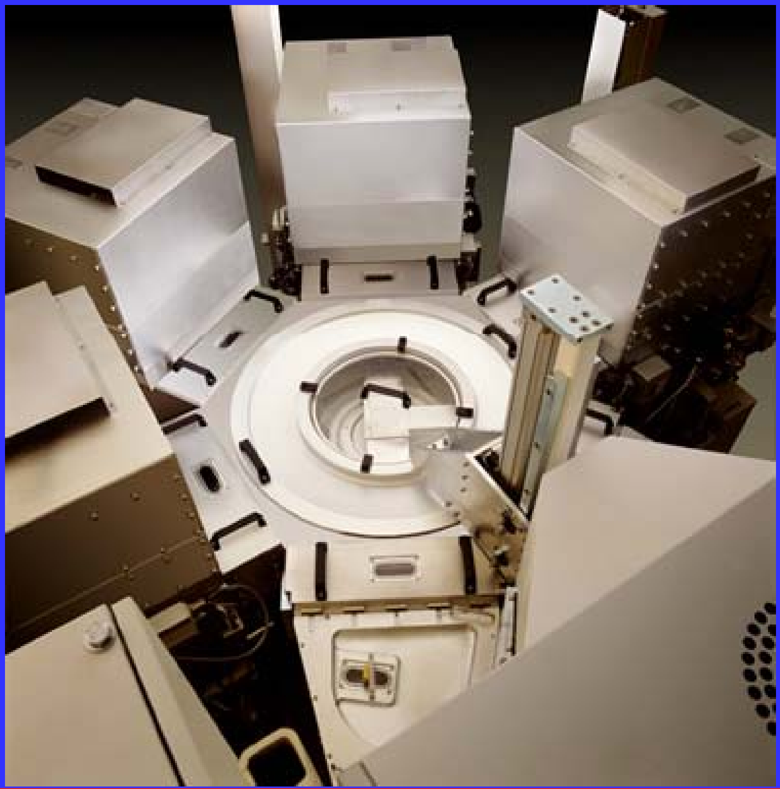
MengChu Zhou

New Jersey Institute of Technology

and

XiDian University

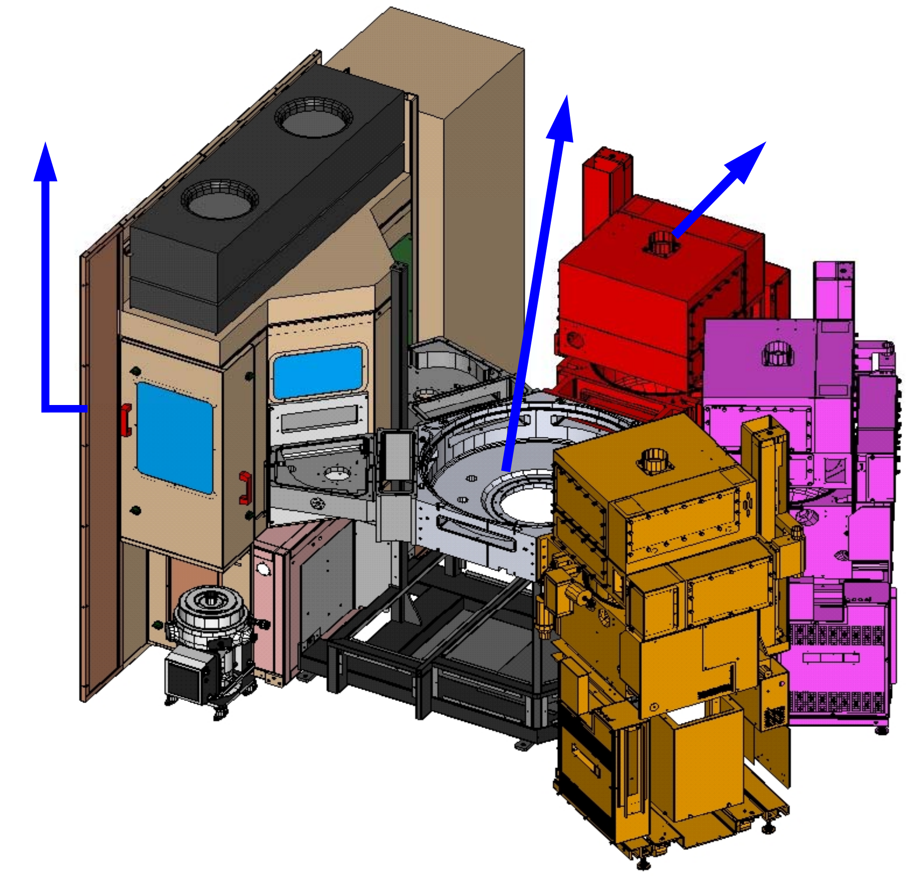
zhou@njit.edu



NJIT New Jersey's Science & Technology University

Discrete Event Systems Laboratory
Department of Electrical and Computer Engineering

Cluster Tools for Semiconductor Manufacturing



- An integrated manufacturing system
  - ➢ process module (PM)
  - ➢ transport module (TM)
  - ➢ cassette module (CM)
- Mechanically interconnected

Why Use Cluster Tools?

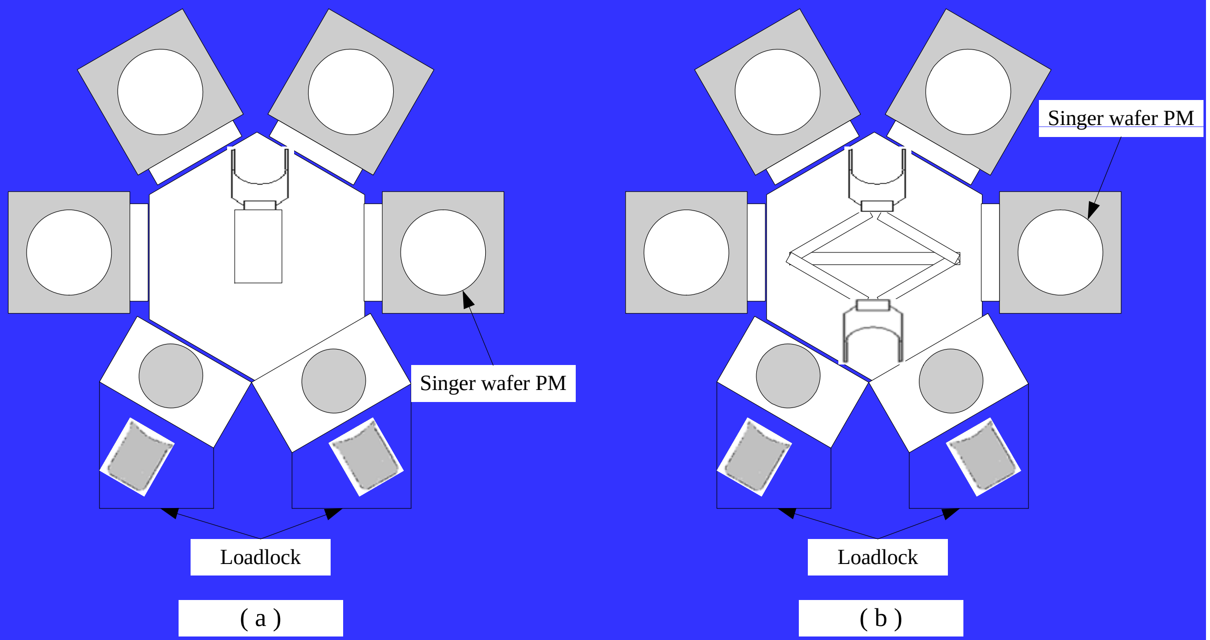
- Highly compact
- Flexibility
- Extendibility and scalability

→

- High throughput
- Multi-type products
- Low cost

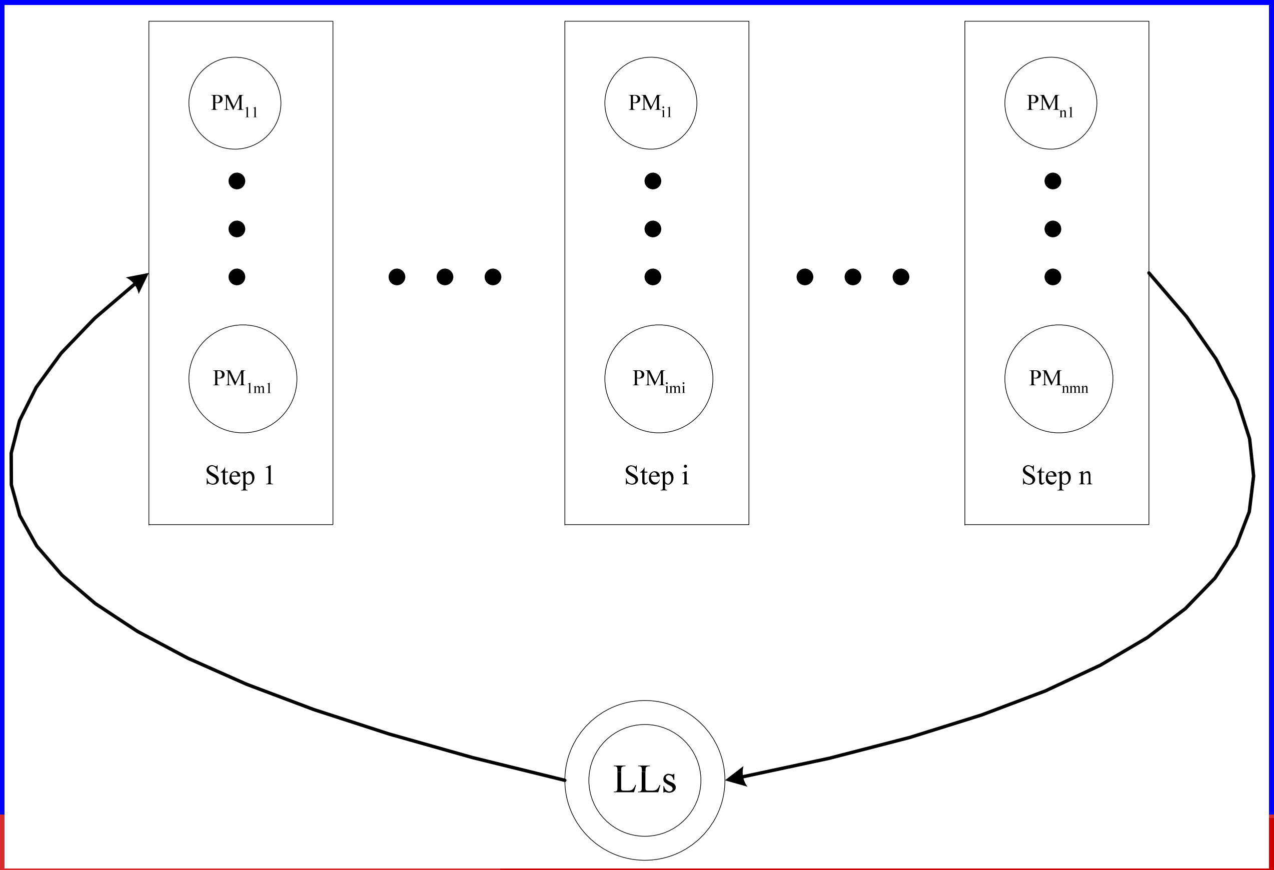
Cluster Tools

- Configuration
  - A robot
    - Single arm
    - Dual arm
  - Process modules (PM)
  - Loadlocks (LL)
  - No intermediate buffer



Wafer Flow Pattern

- Single wafer type
- n steps (operations)
- $(m_1, \ldots, m_i, \ldots, m_n)$: m_i parallel PMs for Step i

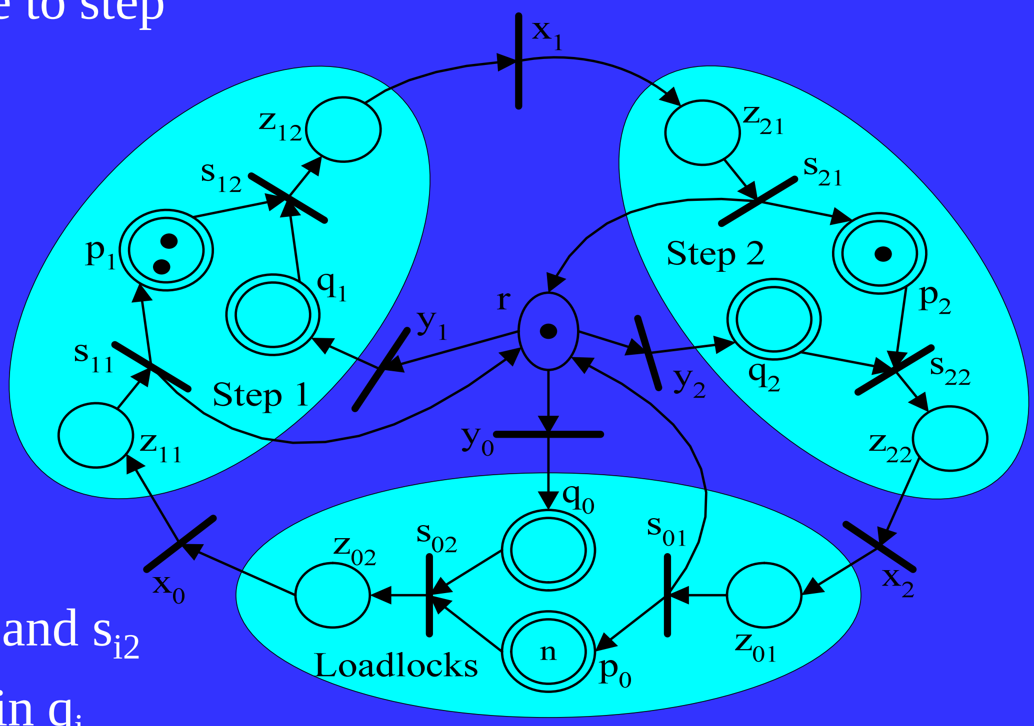


Scheduling Problem

- To schedule the production cycle with minimal cycle time
- The problems
  - Residency time constraint: after being completed, a wafer should be unloaded from a PM in a given time
  - Activity time variation
  - Schedulability
  - Scheduling algorithm
- Key: robot activity scheduling including robot waiting

Petri Net Modeling of Single Arm Cluster Tools (SACT)

- p_i: wafer processing at step i
- x_0: unload wafer from LL and move to step 1
- x_i: move from step i to i+1
- x_n: return wafer from step n to LL
- y_i: move from step i+2 to i
- y_{n-1}: move from step 0 to n-1
- y_n: move from step 1 to n
- Robot waiting at p_i: q_i
- Load and unload from and to p_i: s_{i1} and s_{i2}
- Scheduling: determining wait time in q_i



Time Modeling of SACT

Symbol	Transition or place	Actions	Allowed time duration
λ	s_{i1}	Robot loads a wafer into step i, $i \in \mathbf{\Omega}$	
λ	s_{i2}	Robot unloads a wafer from step i, $i \in \mathbf{N_n}$	
λ_0	s_{02}	Robot unloads a wafer from a loadlock and aligns it	
μ	x_i	Robot moves from step i to step i+1, i $\mathbf{N_{n-1}}$	
μ	x_n	Robot moves from step n to step 0	
μ	y_i	Robot moves from step i+2 to step i, $i \in \mathbf{N_{n-2}}$	
μ	y_{n-1}	Robot moves from step 0 to step n-1	
μ	y_n	Robot moves from step 1 to step n	
τ_i	p_i	A wafer being processed and waiting in p_i, $i \in \mathbf{N_n}$	$[\alpha_i, \alpha_i + \delta_i]$
ω_i	q_i	Robot waits before unloading a wafer from step i, $i \in \mathbf{\Omega}$	$[0, \infty]$
	z_{ij}	No robot activity is associated	0

Wafer Cycle Time of SACT

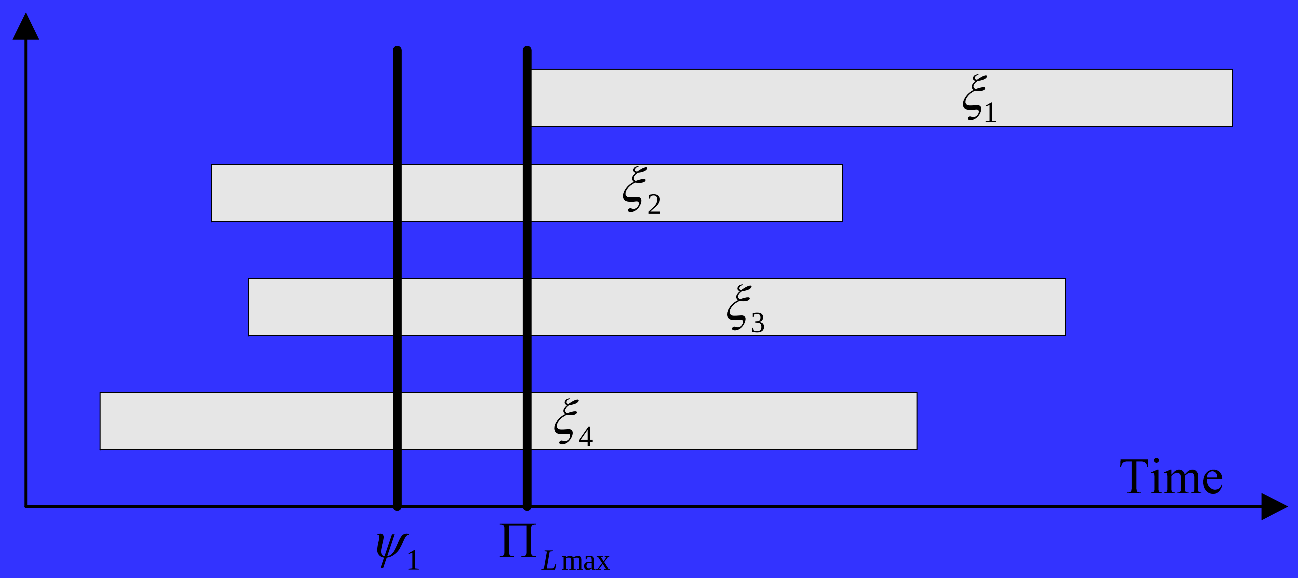
- ξ_i: the time for completing one wafer at step i (cycle time)
- The interval of $\xi_i = [\theta_i, \beta_i]$ determined by residency time constraint for step i
  - $\theta_1 = (\alpha_1+3\lambda+3\mu + \lambda_0)/m_1$
  - $\beta_1 = (\alpha_1+3\lambda+3\mu + \lambda_0 + \delta_1)/m_1$
  - $\theta_i = (\alpha_i+4\lambda+3\mu)/m_i$, $i = 2, \ldots, n$
  - $\beta_i = (\alpha_i+4\lambda +3\mu + \delta_i)/m_i$, $i = 2, \ldots, n$
  - α_i = processing time for step i
  - λ = wafer load/unload time
  - δ_i = the longest time delay after processed
  - m_i = the parallel PMs for step i
- Cycle time $\xi = \xi_i = \xi_j$

Robot Cycle Time of SACT

- Robot cycle time $\psi = \psi_1 + \psi_2$ where
  - $\psi_1 = 2(n+1)\mu + (2n+1)\lambda + \lambda_0$
  - $\psi_2 = \boxed{\sum_{i=0}^{n+1} \omega_i}$
  - μ = time for moving from a step to another
  - λ = time for loading a wafer to a PM, or unloading from a PM
  - λ_0 = time for unloading a wafer from a LL and aligning it
  - ω_i = robot waiting time at step i
- $\psi = \xi$

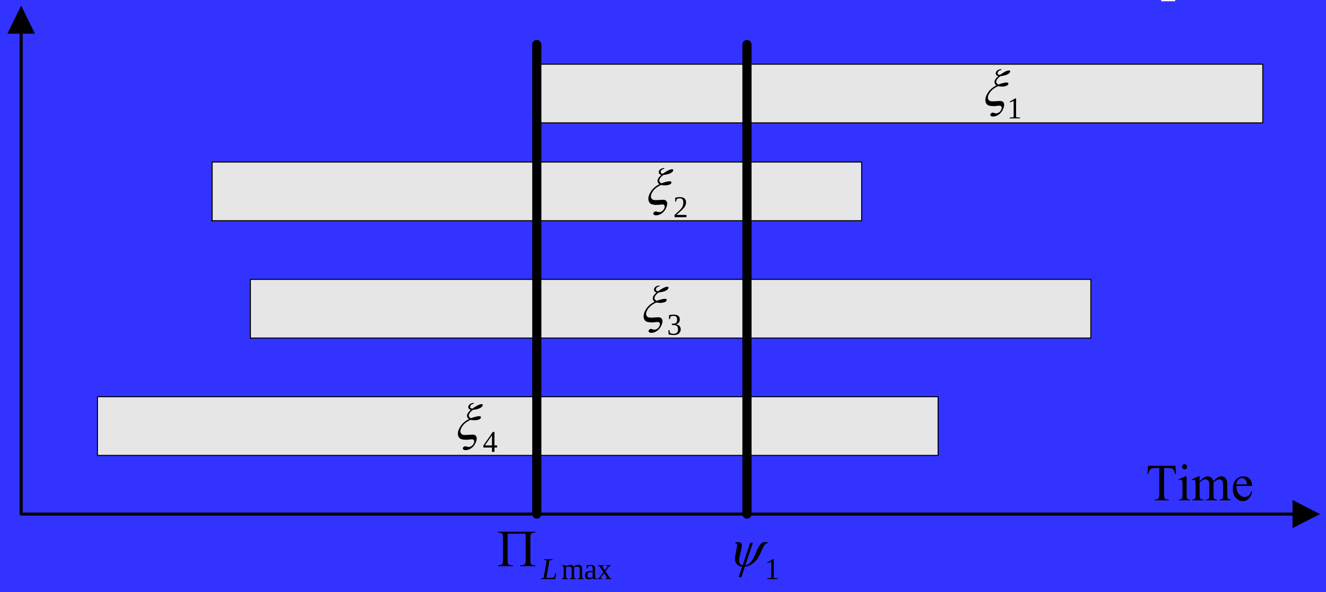
Schedulable Case 1 of SACT

- $\cap_i \xi_i \neq \varnothing$: the intersection of feasible time interval for all steps is not empty
- $\psi_1 \leq \prod_{Lmax}$: in a cycle, the robot has idle time



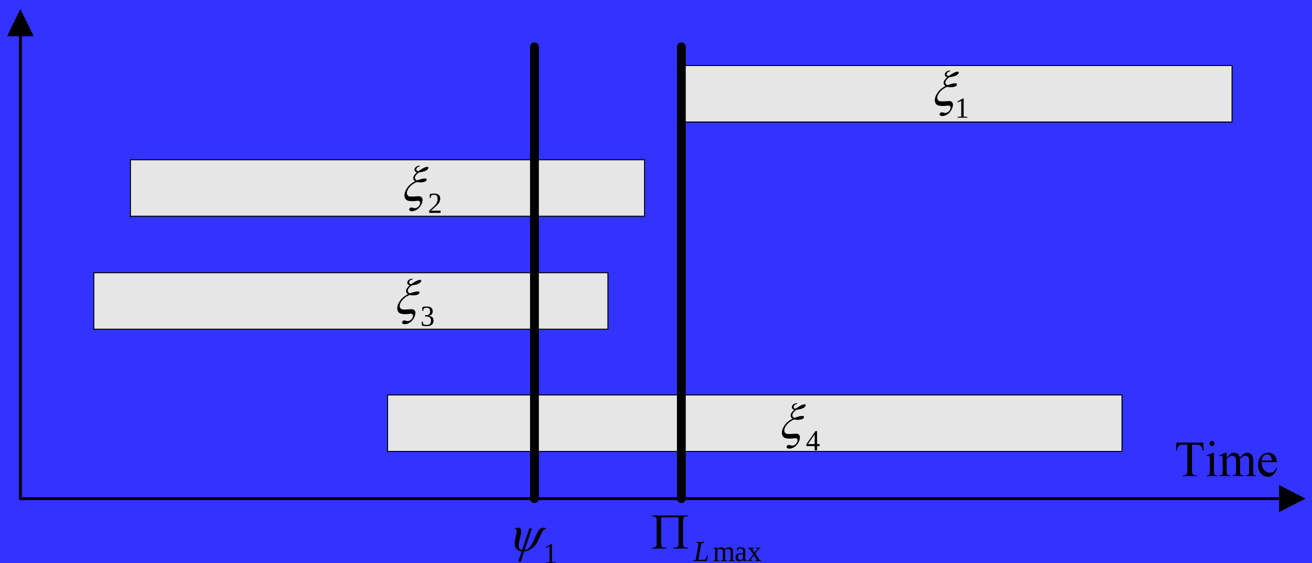
Schedulable Case 2 of SACT

- $\cap_i \xi_i \neq \varnothing$: the intersection of feasible time interval for all steps is not empty
- Transportation-bounded, but the robot cycle is in the intersection of feasible time interval for all steps



Schedulable Case 3 of SACT

- $\cap_i \pi_i = \varnothing$: the intersection of feasible time interval for all steps is empty
- $\psi_1 < \prod_{L\max}$: robot has enough idle time for waiting before unloading wafers at some steps



Any other case is unschedulable.

Scheduling Algorithm of SACT

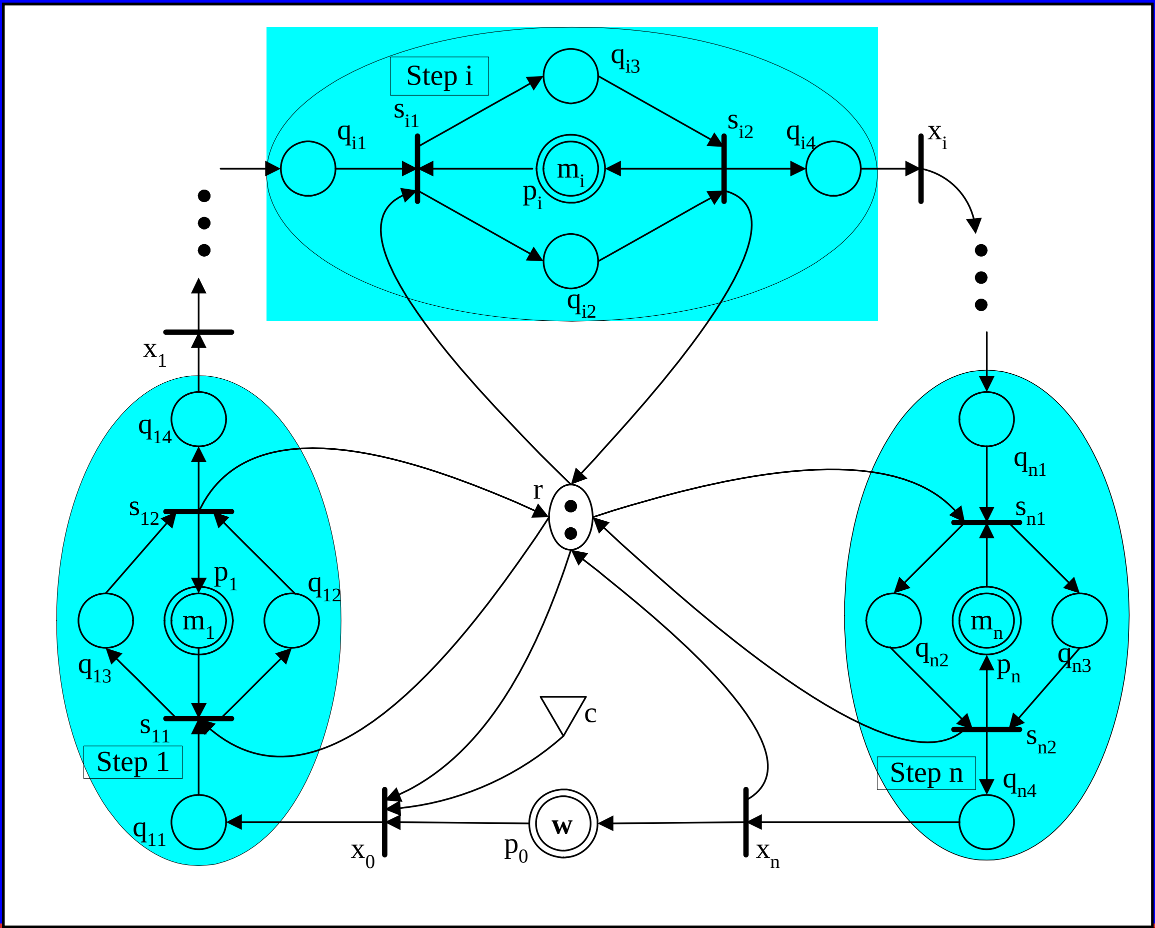
- Case 1: simply set $\omega_n = \psi_2 = \Pi_{Lmax} - \psi_1$
- Case 2: simply set $\omega_k = 0$, $k = 0, \ldots, n$
- Case 3: $\psi_2 = \psi_{21} + \psi_{22}$, $\psi_{21} = \omega_0 + \ldots + \omega_{n-1}$, and $\psi_{22} = \omega_n$,

$$\omega_{i-1} = \begin{cases} 0, i \in F \\ m_i \times \Pi_{L\max} - \alpha_i - \delta_i - 4\lambda - 3\mu, i \in E \cap \{2,3,\ldots,n\} \\ m_1 \times \Pi_{L\max} - \alpha_1 - \delta_1 - 3\lambda - \lambda_0 - 3\mu, i \in E \cap \{1\} \end{cases}$$

- The schedule is optimal in terms of cycle time
- It is a closed-form algorithm

Petri Net Modeling for Dual Arm Cluster Tools (DACT)

- Robot wait: q_{i3}
- Swap: s_{i1} and s_{i2}
- Wait during swap: q_{i2} and q_{i3}
- Scheduling: determining wait time in q_{ij}



Time Modeling of DACT

Symbol	Transition or place	Actions	Allowed time duration
μ_0	x_0	Robot unloads a wafer from a loadlock and moves to p_1	
$\mu_i = \mu$	x_i	Robot moves from p_i to p_{i+1}, $i \in \mathbf{N}_{n-1}$	
μ_n	x_n	Robot moves to a loadlock and returns a wafer	
τ_i	p_i	A wafer being processed and waiting in p_i, $i \in \mathbf{N}_n$	$[a_i, a_i + \delta_i]$
λ	s_{i1} and s_{i2}	Simple swap operation at p_i, $i \in \mathbf{N}_n$	
ω_i	q_{i2} or q_{i3}	Robot waits during swap at p_i, $i \in \mathbf{N}_n$	$[0, \gamma]$
ω_{i1}	q_{i1}	Robot waits at p_i, $i \in \mathbf{N}_n$	$[0, \infty]$
ω_{i4}	q_{i4}	The end of swap operation at p_i, $i \in \mathbf{N}_n$	0

Wafer Cycle Time of DACT

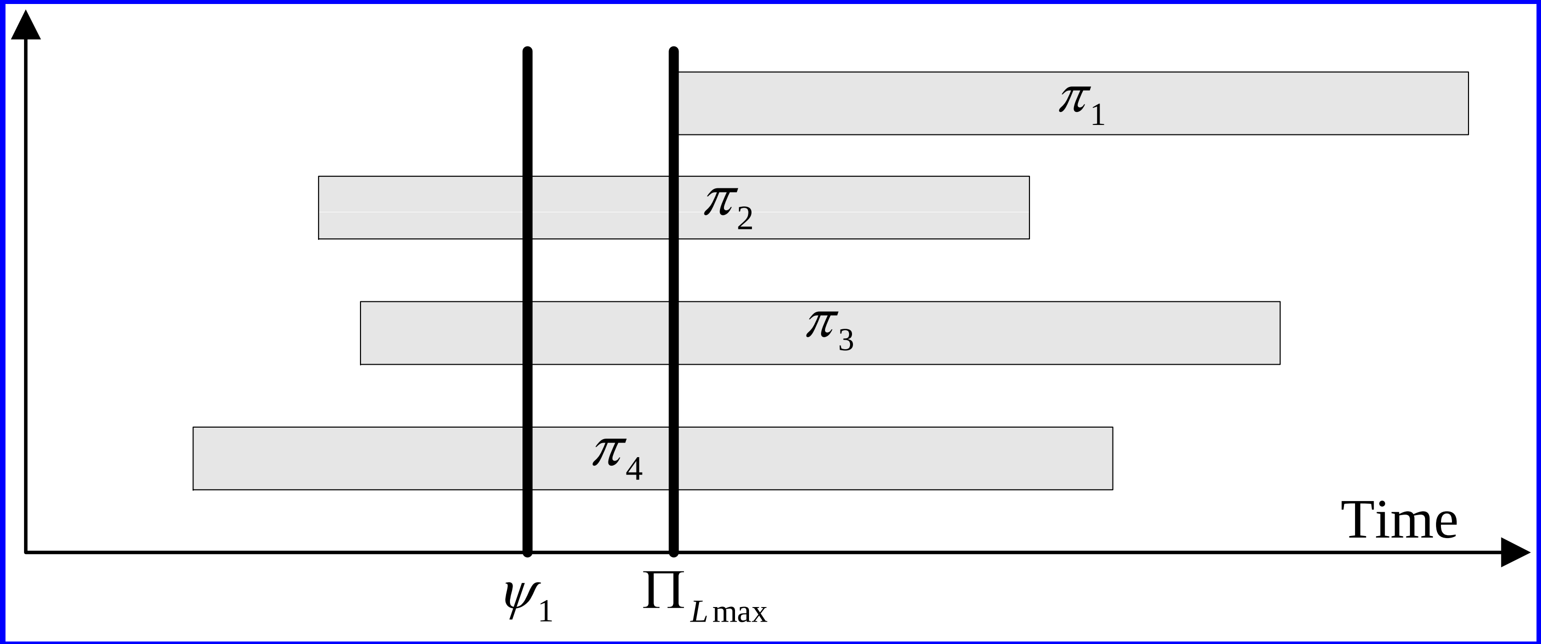
- π_i: the time for completing one wafer at step I (cycle time)
- The interval of $\pi_i = [\alpha_i, \beta_i]$ determined by residency time constraint for step i
  - $\alpha_i = (a_i + \lambda)/m_i$
  - $\beta_i = (a_i + \lambda + \delta_i)/m_i$
  - a_i = processing time for step i
  - λ = swapping time
  - δ_i = the longest time delay after processed
  - m_i = the parallel PMs for step i
- Cycle time $\pi = \pi_i = \pi_j$

Robot Cycle Time of DACT

- Robot cycle time $\psi = \psi_1 + \psi_2$ where
  - $\psi_1 = n \times\lambda + \sum_k \mu_k$
  - $\psi_2 = \sum_k(\omega_{k1} + \omega_k))$
  - μ_0 = time for unloading a wafer from an LL
  - μ_i = move from step i to step i+1
  - μ_n = return a wafer to an LL
  - λ = time for swapping
  - ω_{k1} = time for waiting at place q_{k1}
  - ω_k = time waiting during swapping at step k
- $\psi = \pi$

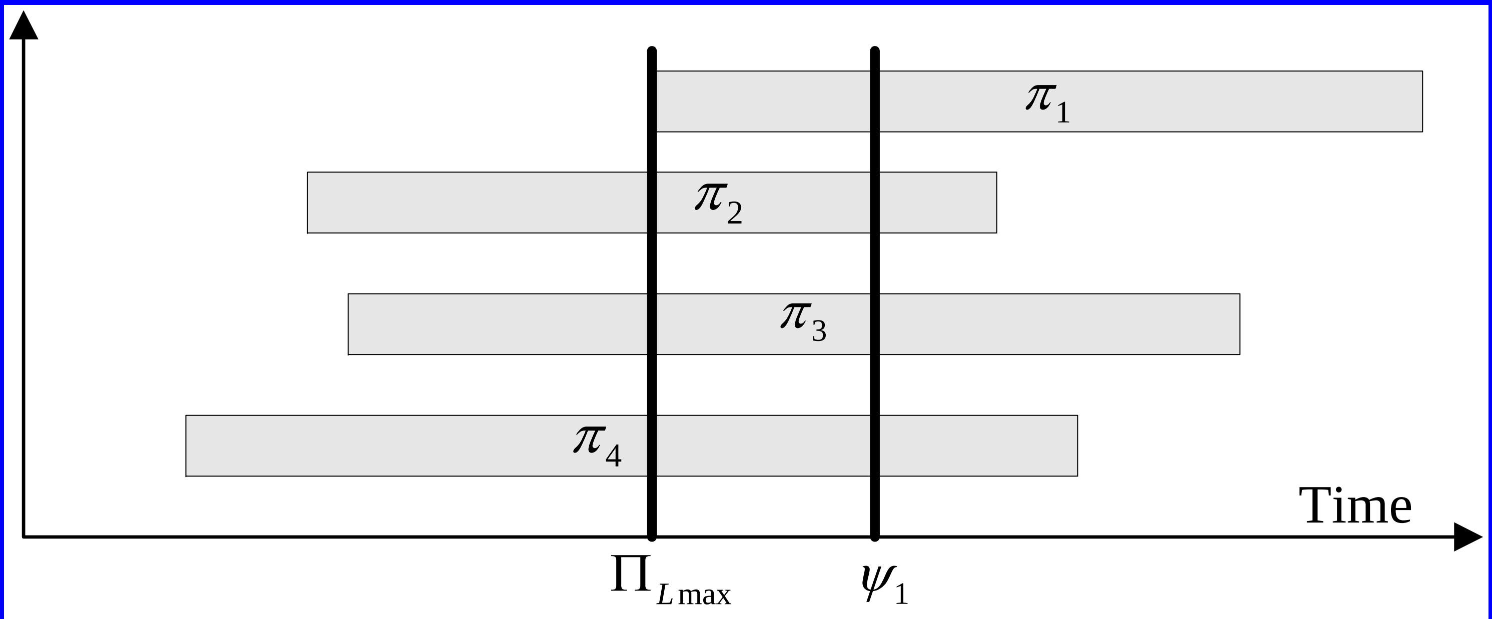
Schedulable Case 1 of DACT

- Non-empty intersection of feasible time intervals for all steps
- Robot has idle time in a cycle
- The feasible time interval and robot cycle are different from single arm cluster tools



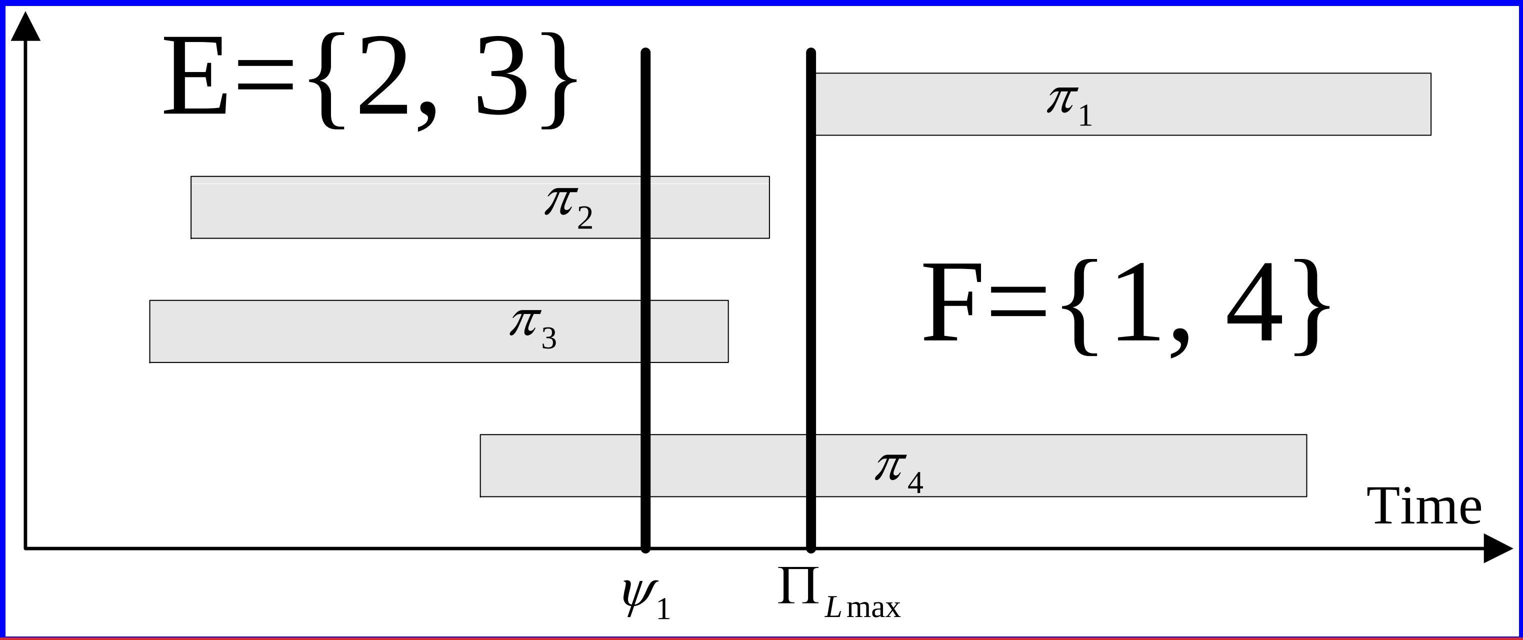
Schedulable Case 2 of DACT

- Non-empty intersection of feasible time intervals for all steps
- Transportation-bounded, but the robot cycle is in the intersection of feasible time interval for all steps
- The feasible time interval and robot cycle are different from single arm cluster tools



Schedulable Case 3 of DACT

- Non-empty intersection of feasible time intervals for all steps
- Robot has enough idle time for waiting before unloading wafers at some steps
- The feasible time interval and robot cycle are different from single arm cluster tools



Scheduling Algorithm

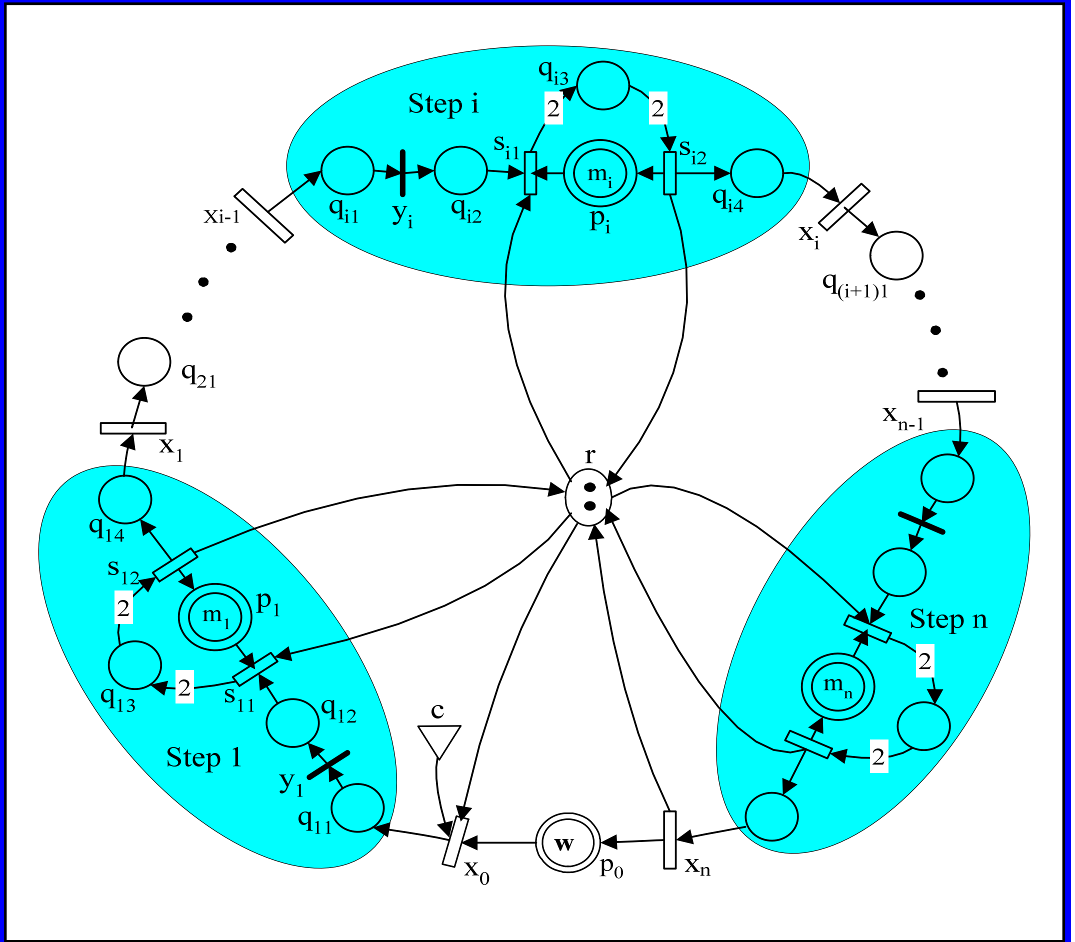
- Case 1: simply set $\psi_2 = \Pi_{Lmax} - \psi_1$
- Case 2: simply set $\psi_2 = 0$
- Case 3: $\psi_2 = \psi_{21} + \psi_{22}$, ψ_{21} is for waiting during swap operations, and ψ_{22} is for other waiting
- $\psi 21 = \boxed{\sum_{k=1}^{n} \omega_k}$ $\psi_{22} = \boxed{\sum_{k=1}^{n} \omega_{k1}}$

$$\boxed{\omega_i = \begin{cases} 0, i \in F \\ m_i \times \Pi_{L\max} - a_i - \delta_i - \lambda, i \in E \end{cases}}$$

- The schedule is optimal in terms of cycle time
- It is an analytical algorithm

Modeling DACT with Activity Time Variation

- p_i: wafer processing at step i
- x_0: unload wafer from LL and move to step 1
- x_i: move from step i to i+1
- x_n: return wafer from step n to LL
- q_{i1}: scheduled robot wait
- q_{i2}: unscheduled robot wait
- Swap: s_{i1} and s_{i2}
- Robot wait during swap: q_{i3}
- c: control place
- Scheduling: determining robot wait time in q_{ij}



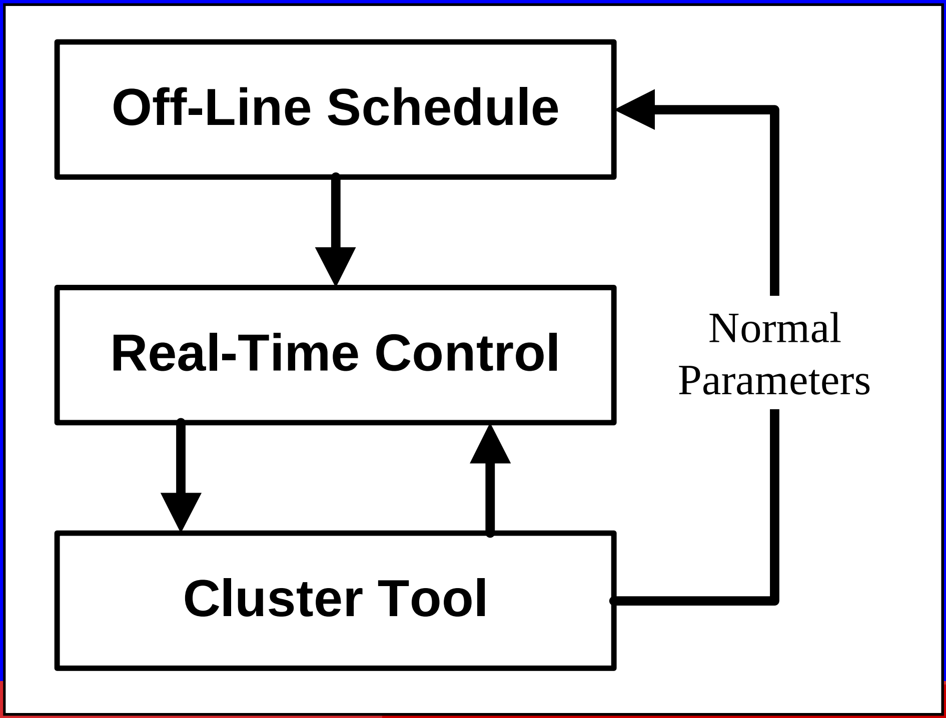
Time Modeling

[A, B] means that, at normal condition, a task takes time A and can vary up to B.

Symbol	Transition or place	Actions	Allowed time duration
μ_0	x_0	Robot unloads a wafer from a loadlock and moves to p_1	$[Q_0, S_0]$
μ_i	x_i	Robot moves from p_i to p_{i+1}, $i \in \mathbf{N}_{n-1}$	[Q, S]
μ_n	x_n	Robot moves to a loadlock and returns a wafer	$[Q_n, S_n]$
τ_i	p_i	A wafer being processed and waiting in p_i, $i \in \mathbf{N}_n$	$([A_i, B_i], \delta_i]$
λ_i	s_{i1} and s_{i2}	Simple swap operation at p_i, $i \in \mathbf{N}_n$	[C, D]
ω_i	q_{i3}	Robot waits during swap at p_i, $i \in \mathbf{N}_n$	$[0, \Gamma]$
ω_{i1}	q_{i1}	Scheduled robot waits at p_i, $i \in \mathbf{N}_n$	$[0, \infty]$
α_{i2}	q_{i2}	Unscheduled robot waits at p_i, $i \in \mathbf{N}_n$	$[0, \infty]$
ω_{i4}	q_{i4}	The end of swap operation at p_i, $i \in \mathbf{N}_n$	0

Operation Architecture

- Two levels
  - Upper level: off-line schedule
  - Lower level: real-time control
- Find the off-line schedule under normal condition by determining ω_{i1} and ω_i



Real-Time Controller

- Under normal condition

$\mu_0^j = Q_0$: j stands for an activity's jth occurrence

$\mu_i^j = Q$

$\lambda_i^j = C$

ω_{i1} and ω_i are scheduled to be constants

Real-Time Controller

- In real-time, let

$$\mu_0^j = Q_0 + \sigma_0^j$$

$$\mu_i^j = Q + \sigma_i^j$$

$$\mu_n^j = Q_n + \sigma_n^j$$

$$\lambda_i^j = C + \rho_i^j$$

- Regulate ω_{i1} in real-time as

$$\omega_{11}^j = Max\{0, \omega_{11} - (\sigma_0^j + \sigma_n^j)\}$$

$$\omega_{i1}^j = Max\{0, \omega_{i1} - \sigma_i^j\}$$

Control Policy

1. Under the normal condition, find an off-line periodic schedule by determining ω_{i1} and ω_i, $i \in \mathbf{N}_n$.
2. Transition y_1 is enabled if the jth token stays in q_{11} for
$$\omega_{11}^j = Max\{0, \omega_{11} - (\sigma_0^j + \sigma_n^j)\}$$
3. Transitions y_i, $i \in \mathbf{N}_n$-{1} are enabled if the jth token stays in q_{i1} for $\omega_{i1}^j = Max\{0, \omega_{i1} - \sigma_i^j\}$
4. A token should stay in q_{i3} for , $i \in \mathbf{N}_n$, time units; and
5. Transitions s_{i1} and x_i fire once being enabled

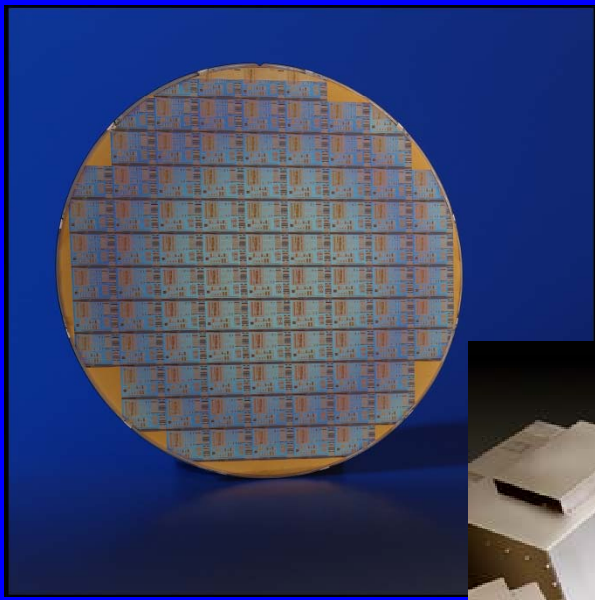
Illustrative Example

- Parameters
  - Flow pattern: (1, 1)
  - $\mu_0 \in [23, 40]$
  - $\mu_1 \in [2, 3]$
  - $\mu_1 \in [14, 16]$
  - $\lambda \in [21, 23]$
  - $A_1 = A_2 = 120$
  - $B_1 = B_2 = 125$
  - $\delta_1 = \delta_2 = 20$
- It is shown that it is not always schedulable
- By our approach, it is always schedulable

Conclusions

- Contribution
  - Compact and reusable PN model
  - A novel operation architecture
  - Schedulability analysis for deterministic activity time
  - Closed-form optimal scheduling algorithms
  - An effective real-time control policy
- Future work
  - real-time schedulability
  - scheduling algorithm with activity time variations
  - Revisiting wafer pattern flow
  - Multiple cluster tools

Questions?



Thank you!