Modular Logic Controllers for Machining Systems: Formal Representation and Analysis using Petri Nets

Dawn Tilbury

Mechanical Engineering and Applied Mechanics

University of Michigan

Acknowledgments

- **Joint work with Euisu Park and Pramod Khargonekar of University of Michigan**
- **Funding provided by NSF/ERC for Reconfigurable Machining Systems at UM**
- **Industrial partners from Lamb Technicon for motivation, examples, and feedback**

Outline

- Motivation: Logic control problem for high-volume machining systems
- Background
- Automatic cycle logic control
- Multi-mode logic control
- Implementation and future work

High-volume machining systems

- Cycle time determined by projected demand
  - 500,000 parts/year ➡ one part every 20 sec.
  - One operation per station
- Dedicated transfer machine:
  - Fixed material handling, part flow
  - Highly accurate and repeatable operations
  - 10-12 stations, buffer at beginning/end
  - 5,000-10,000 I/O points (sensors/actuators)
  - 10-20 transfer machines per transfer line

Control of machining systems

- Continuous variable control (servo, CNC)
  - Positioning, feed for metal removal, fixturing, and transfer operations: mainly SISO loops
- Discrete-variable (Logic) control (PLC)
  - Sequencing of operations
  - Coordination and synchronization of stations
  - User interface (pushbuttons & display)
  - Fault diagnosis and fault handling
  - Safety: gates and interlocks
  - Machine services: coolant, lubrication

Logic control problems

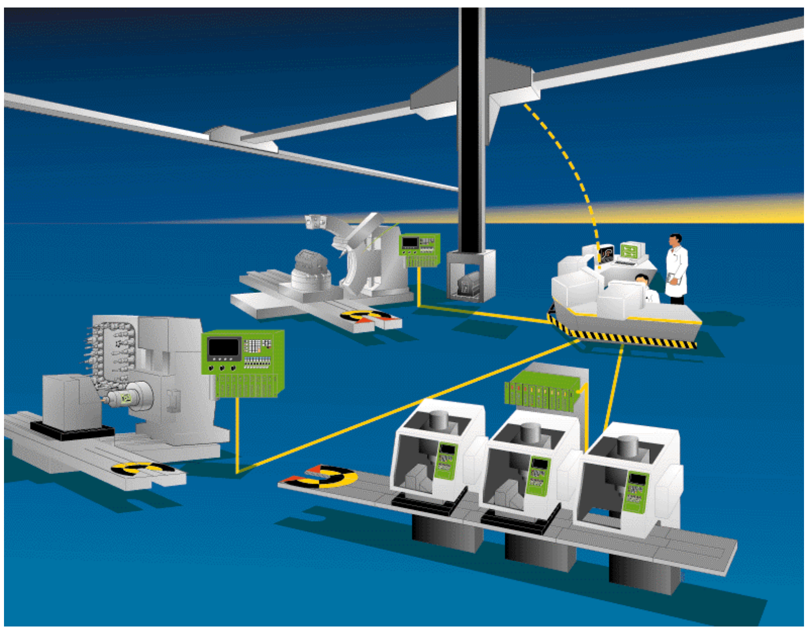
- Control software is 50% of machining system development time and cost
  - Often written in low-level language (ladder)
  - Specification for automatic cycle: timing bar chart, but only requires 10% of control code
  - No specification for alternate control modes, error handling, diagnostics
- No formal verification before implementation
- Long testing and debugging cycles result

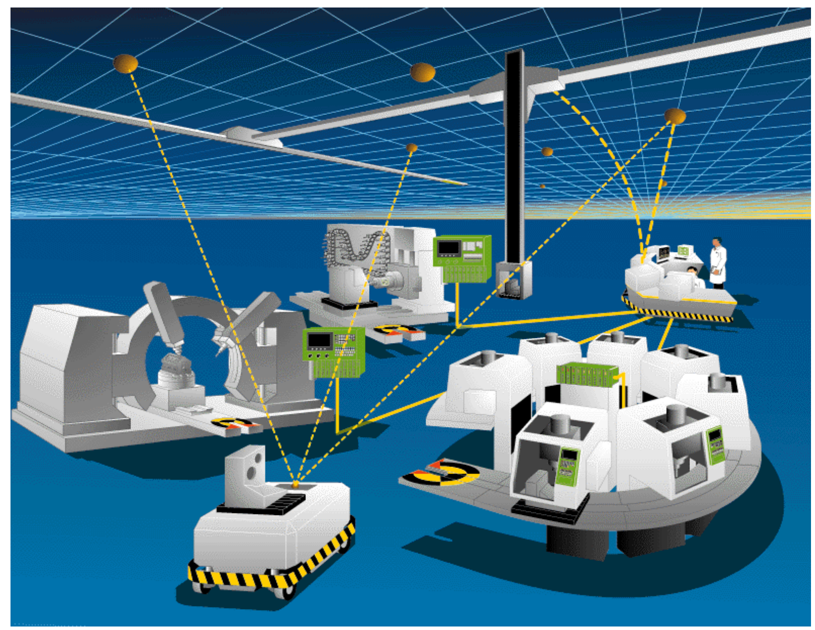
Industry trends

- Faster time to market for new products
  - Must reduce system development time
- Shorter product lifecycles
  - Need to change manufacturing systems to produce new products
- Increased quality
  - Integrate new technologies into existing manufacturing systems

Reconfigurable machining systems

- **Modular: easy assembly, potential reuse**
- **Upgradable: new technology integration**
- **Convertible: quick changeover to new part**





Logic control challenges in RMS

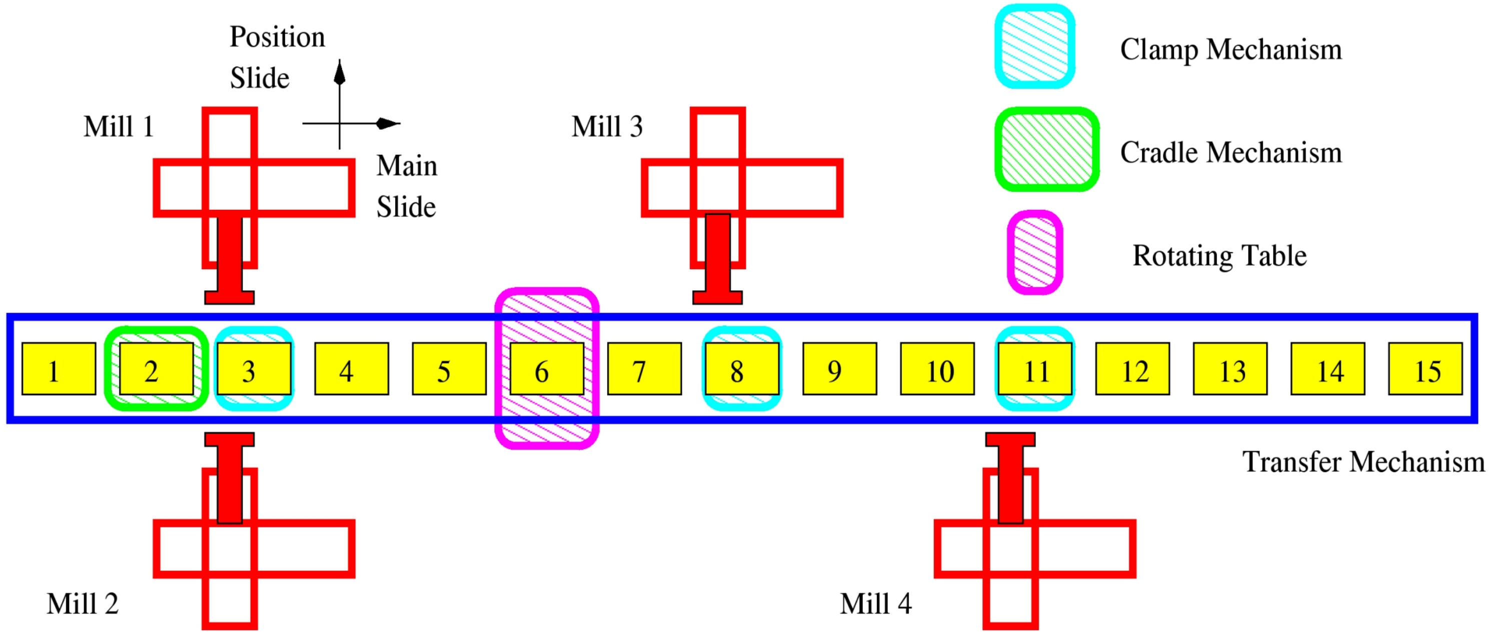
- **Modular structure**
  - **Same granularity as machine modules**
  - **Ease of configuration/reconfiguration**
- **Mathematical basis**
  - **Enable formal verification of correctness**
  - **High-level abstraction for ease of understanding control system functionality**
- **Industrial implementation**
  - **Realistic complexity (thousands of events)**
  - **Target system: PLC or PC code generation**

Outline

- Motivation: Logic control problem for high-volume machining systems
- Background
  - Transfer lines and Petri nets
- Automatic cycle logic control
- Multi-mode logic control
- Implementation and future work

Transfer line example

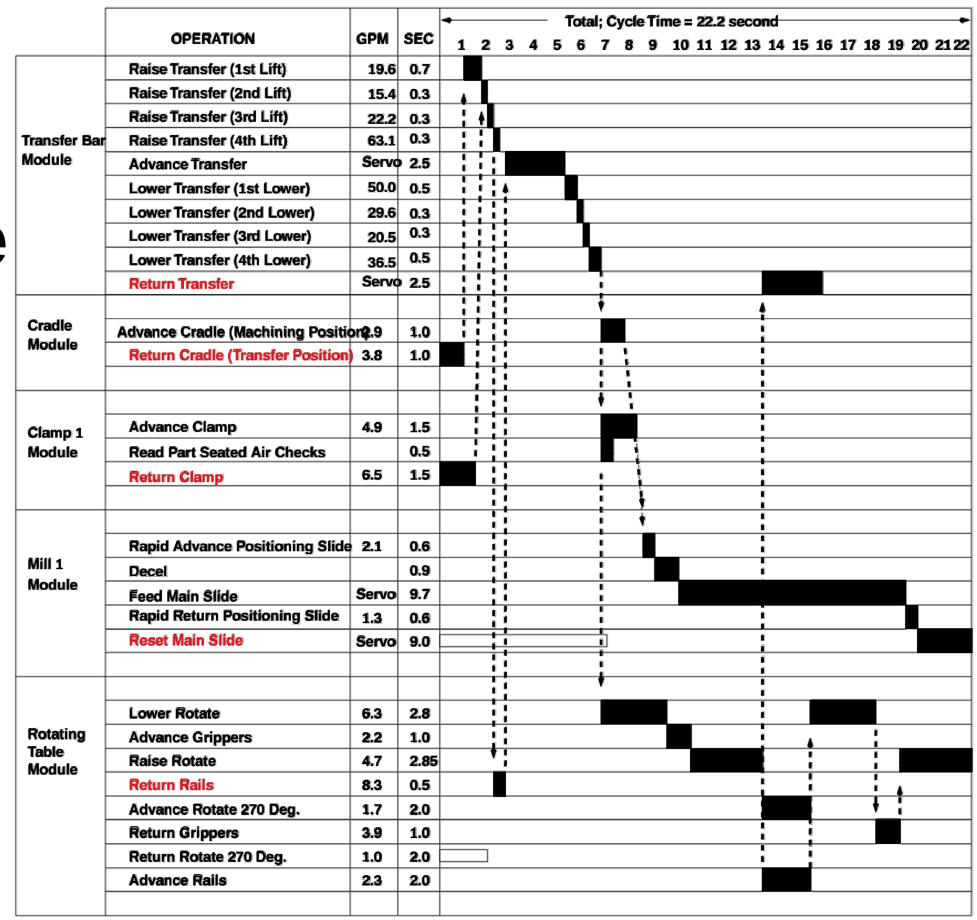
- **Modular, standardized machining systems**
- **Each station self-contained with controls**
- **Coordination through transfer bar**



Control specification

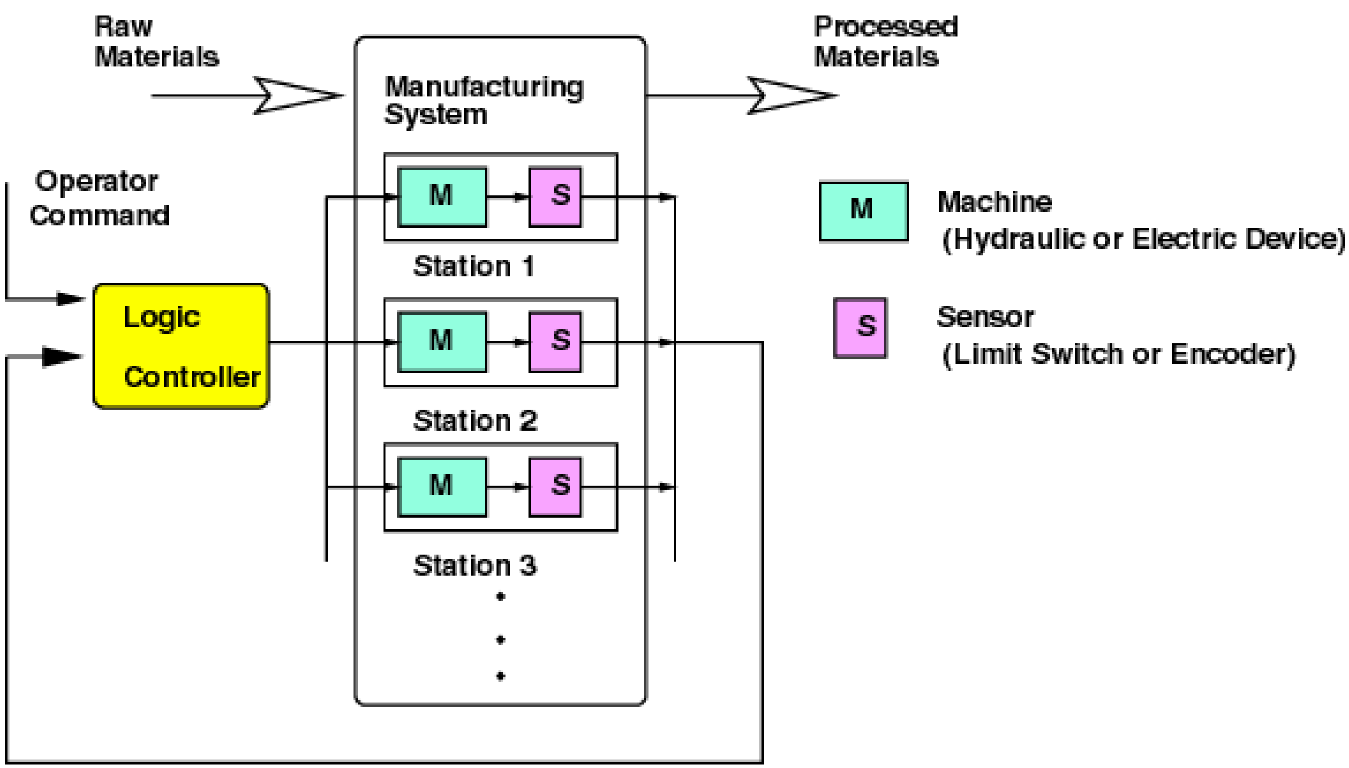
Timing bar chart specifies auto mode

- Developed by machine designers
- Operation sequence
- Causal dependencies
- Home operation
  - Mechanical stability
- Hydraulics
- Cycle time



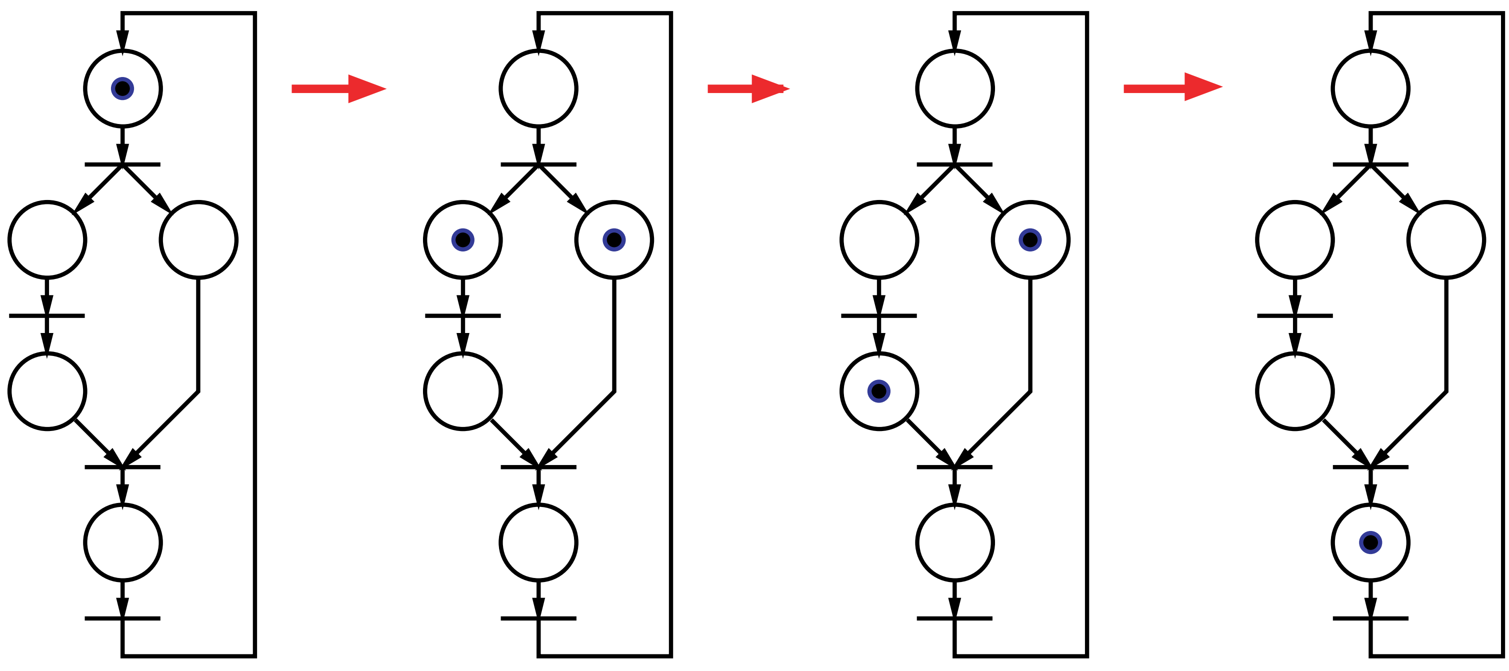
Logic control block diagram

- **Inputs: sensors and operator commands**
- **Outputs: servo or hydraulic devices**



Formal representation of control

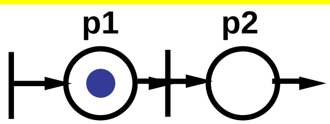
- **Petri net: Bipartite graph**
  - **Places (states) model operations: active or idle**
    - **Token (dot) marks active state**
  - **Transitions model events: from sensors or operator commands**



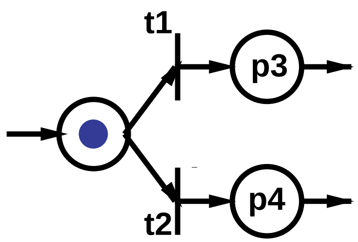
Advantages of Petri net models

- **Large base of existing theory**
- **Verification of key properties of control**
- **Hierarchical structures for complexity management**
- **Implementation in PLC**
  - **SFC = IEC standard programming language**
  - **One-one translation from Petri net to SFC**

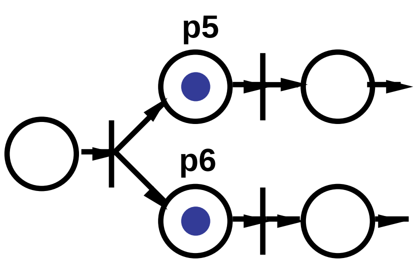
Petri net model capabilities



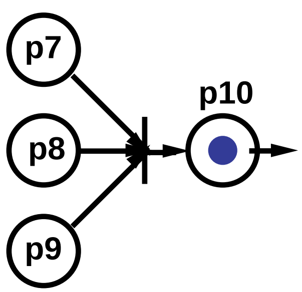
- **Causal dependency**
  - p1 occurs before p2



- **Conflict**
  - Only one of p3, p4 can occur
  - Transition conditions t1, t2 must be mutually exclusive



- **Concurrency**
  - p5, p6 active simultaneously



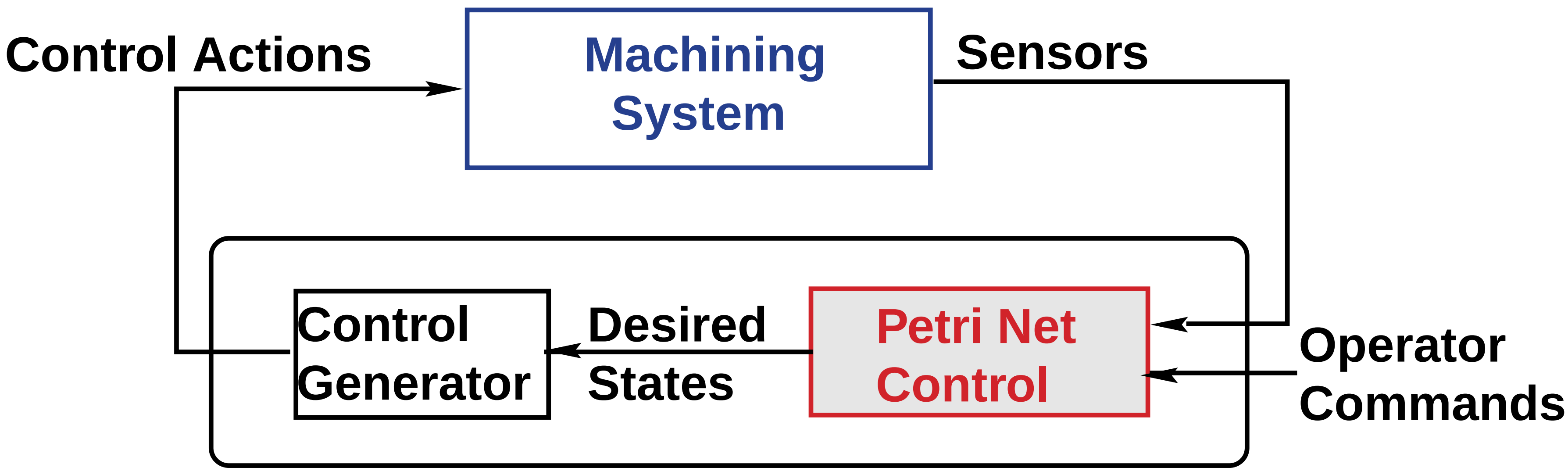
- **Synchronization**
  - p10 cannot begin until p7,p8,p9 have finished

Petri net properties for logic control

- **Live:** every transition can eventually occur
  - No deadlocks
  - All operations and events can happen
- **Safe:** No more than one token per place
  - Ongoing operations are not requested
  - Boolean state representation (active, inactive)
- **Reversible:** Initial state always reachable
  - Guarantees cyclic behavior of system

Supervisory control problem

- Petri net control models desired closed-loop behavior of system
- Control actions generated based on desired state

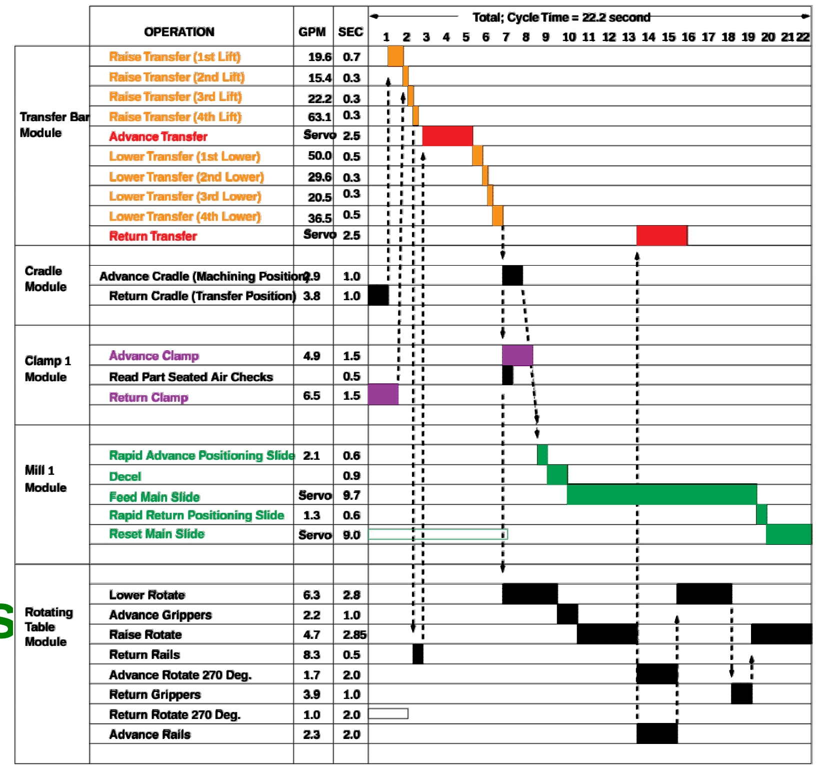


Outline

- **Motivation: Logic control problem for high-volume machining systems**
- **Background**
- **Automatic cycle logic control**
- **Multi-mode logic control**
- **Implementation and future work**

Auto mode

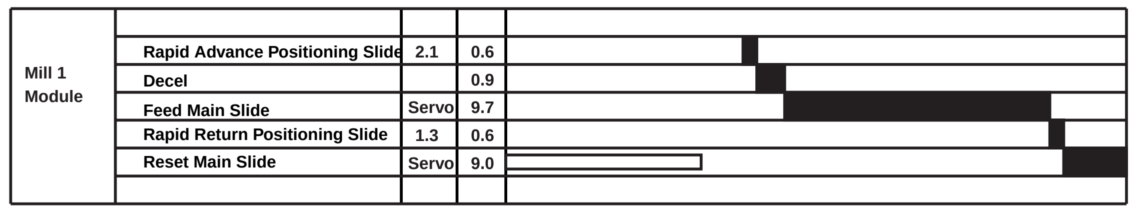
- **Normal operation cycle**
  - Unclamp parts
  - Raise transfer
  - Advance transfer
  - Lower transfer
  - Clamp parts
  - Cycle machining stations
  - Return transfer

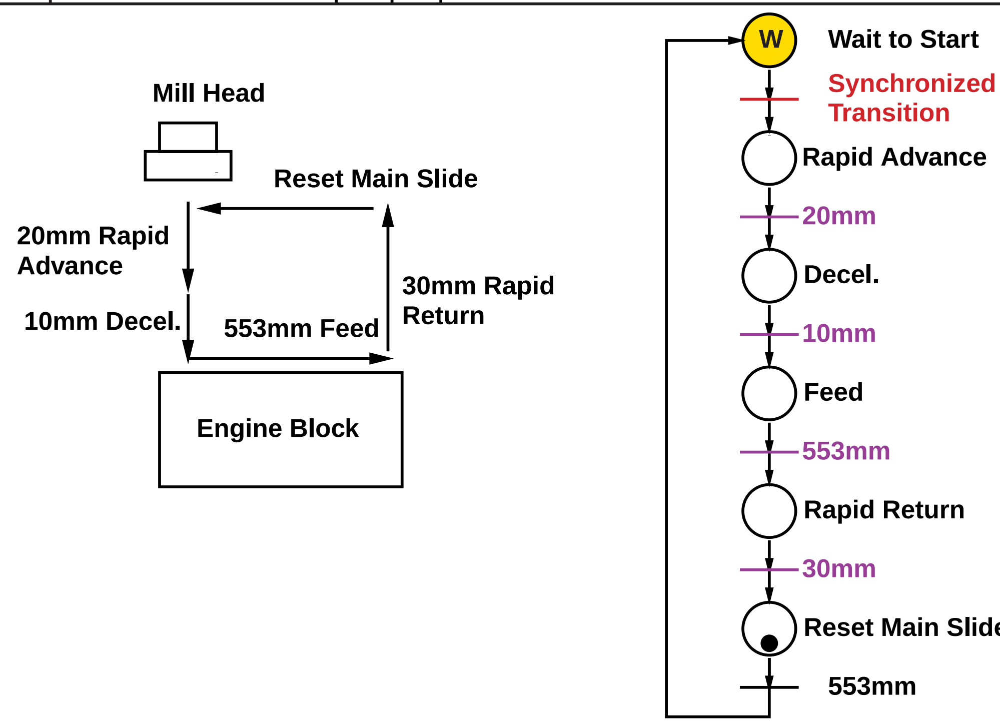


- **Overlapping to minimize cycle time**
- **Fault diagnosis and fault stop**

Timing bar chart -> Petri net

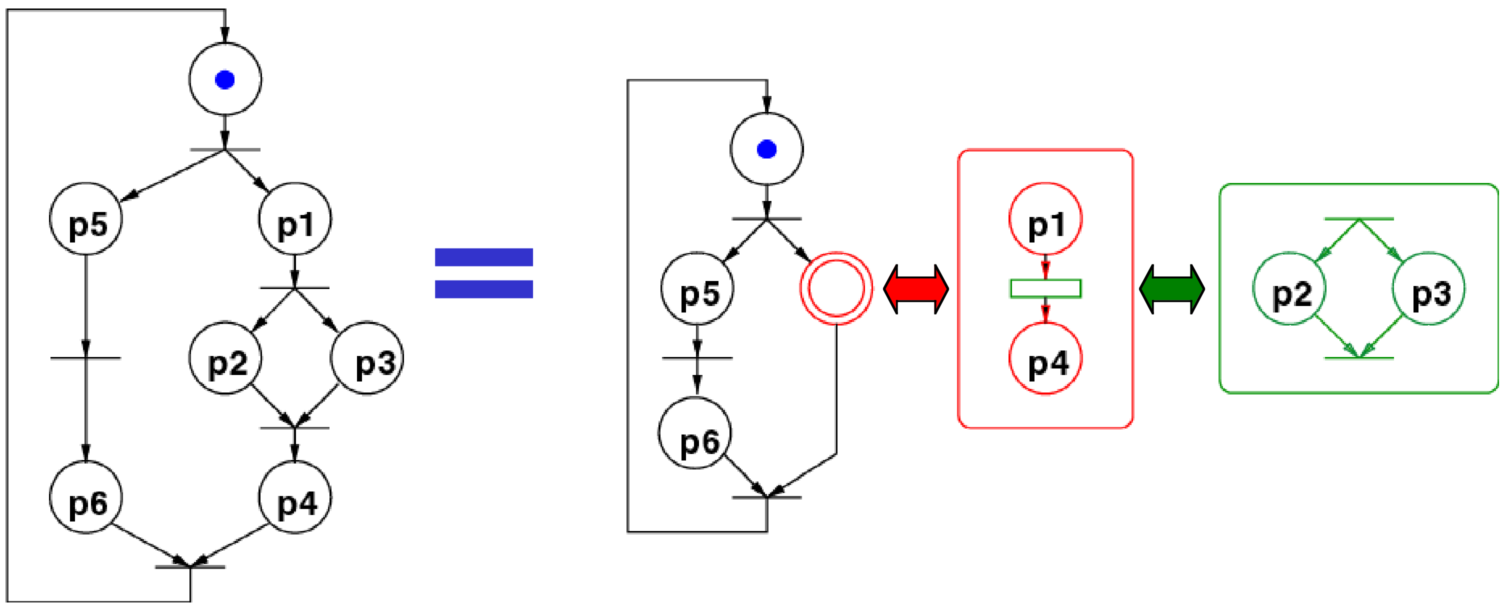
- **Each mechanical module gives one directed circuit**
- **Add wait states before synchronized transitions to model operation termination**





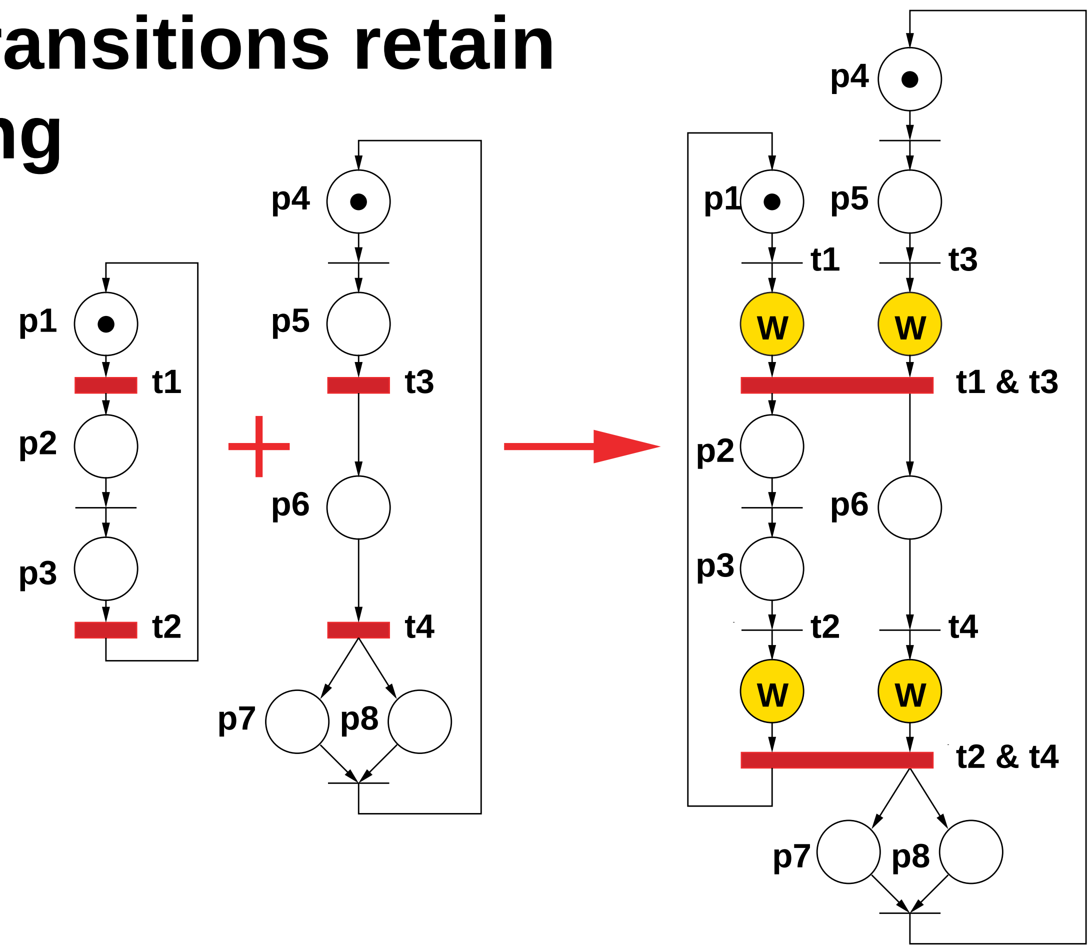
Hierarchical representation

- **Reduce complexity of Petri net by combining sets of places/transitions**
- **Preserves liveness, safeness, reversibility**



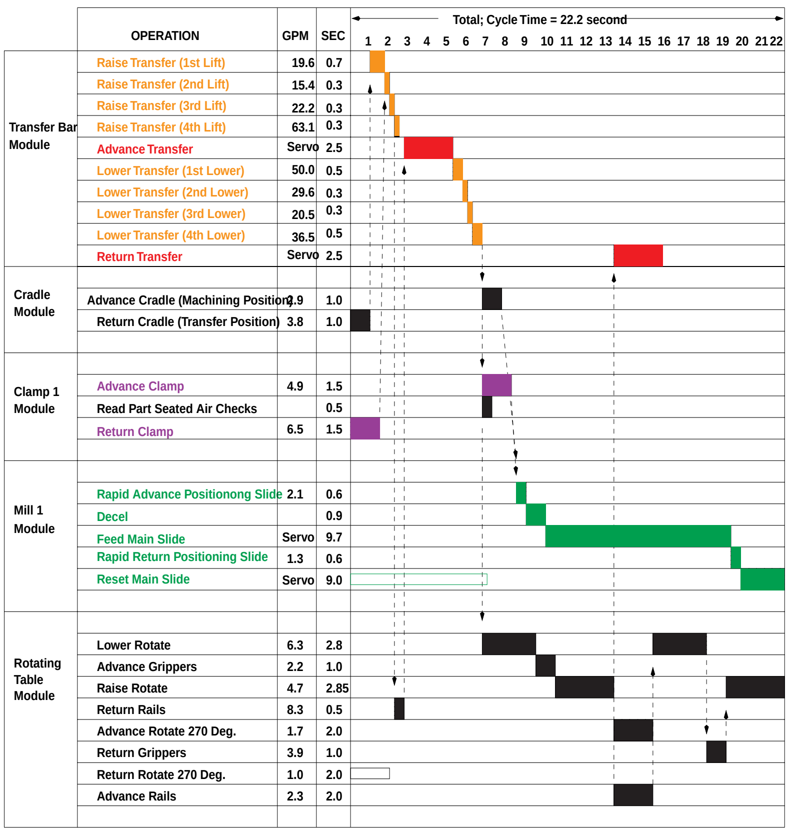
Combining control modules

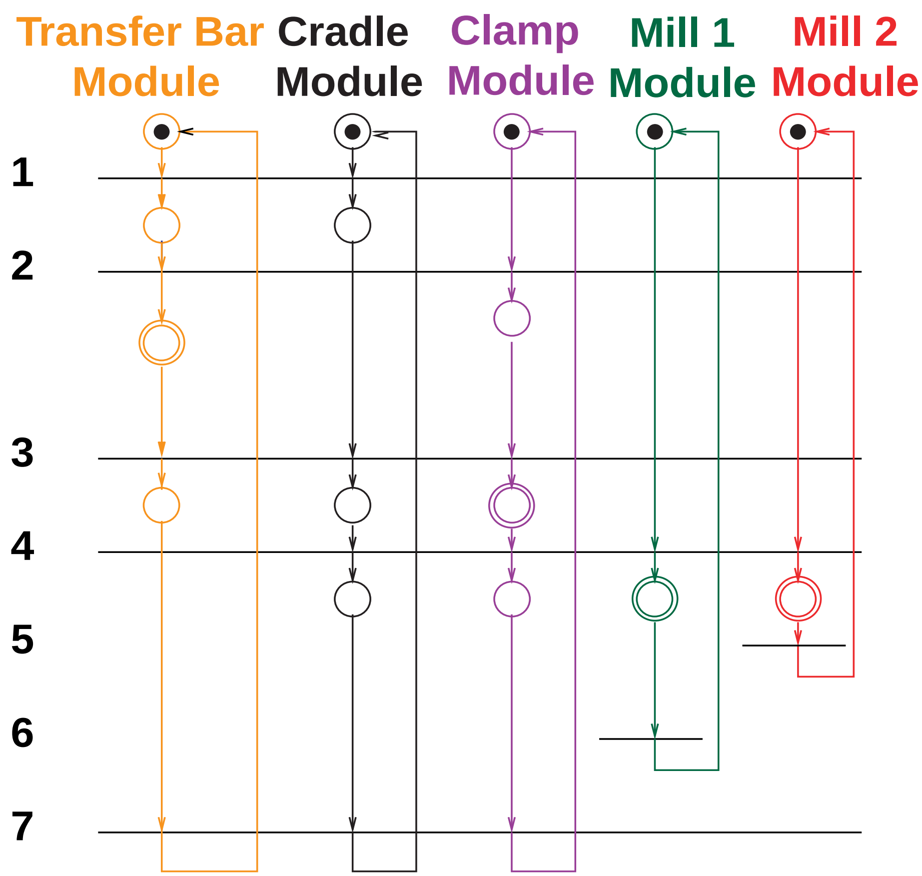
- Merge **synchronized transitions**
- Synchronized transitions retain physical meaning
- Add wait states (W) before synchronized transitions to model operation completion



Logic control for automatic cycle

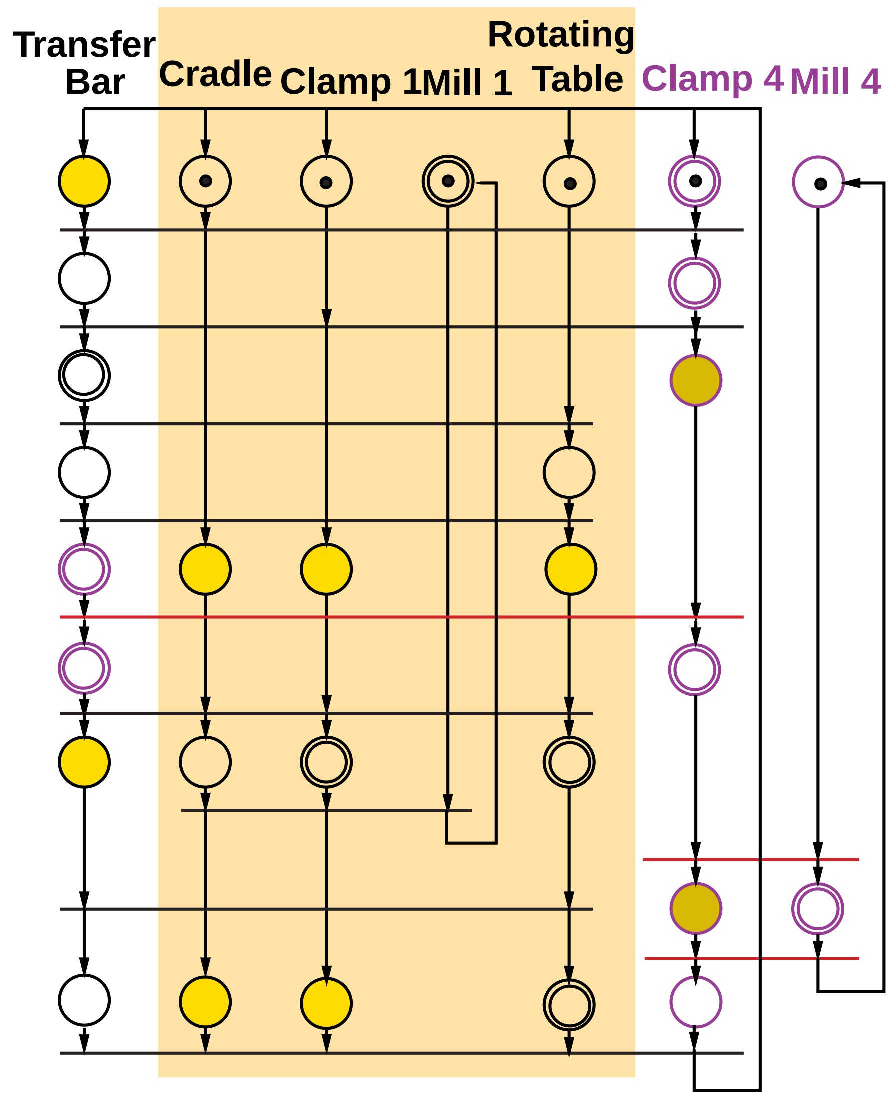
- **Directly generated from timing bar chart**
- **Guaranteed to be live, safe, reversible**





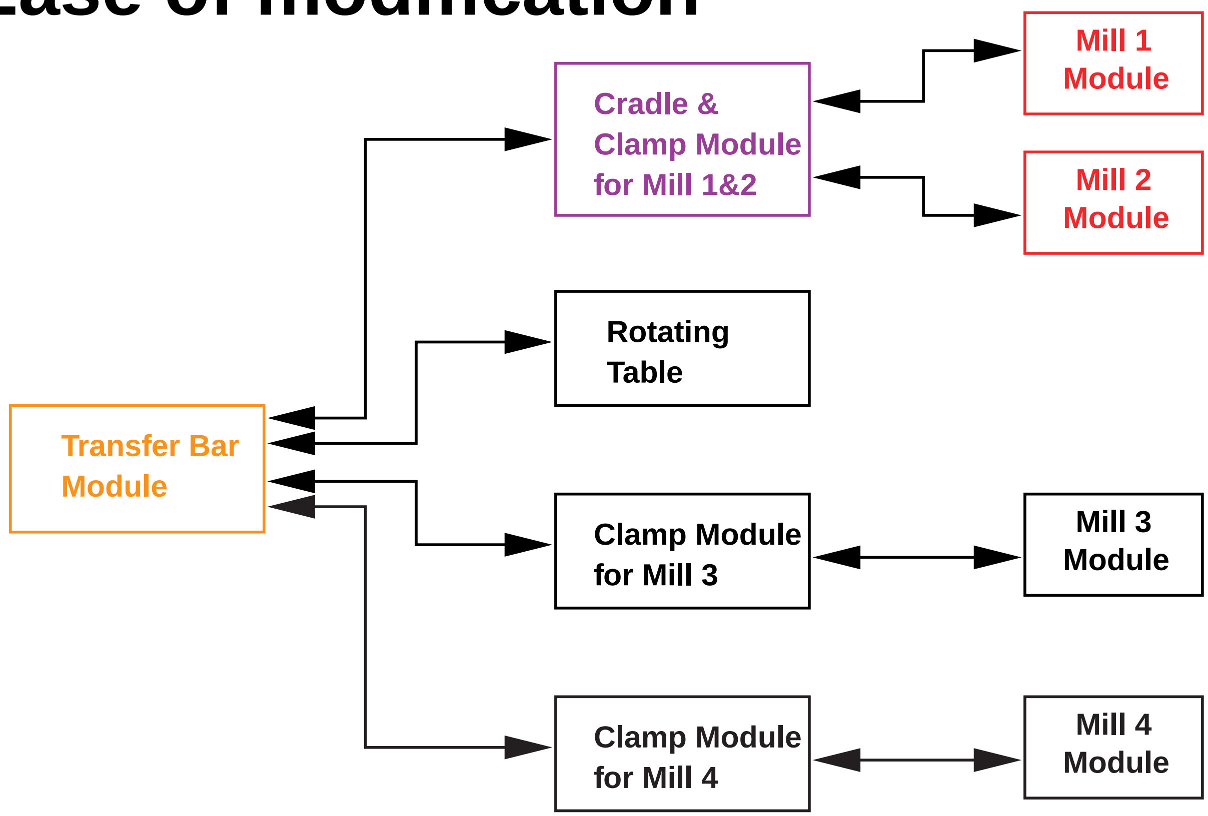
Reconfiguration of logic control

- **Add a new mill to transfer line**
- **New logic for mill & clamp**
- **Modify transfer bar logic to synchronize with clamp**
- **Shaded logic unchanged**



Modular structure of control

- Transfer bar synchronizes all modules
- Limited communication
- Ease of modification

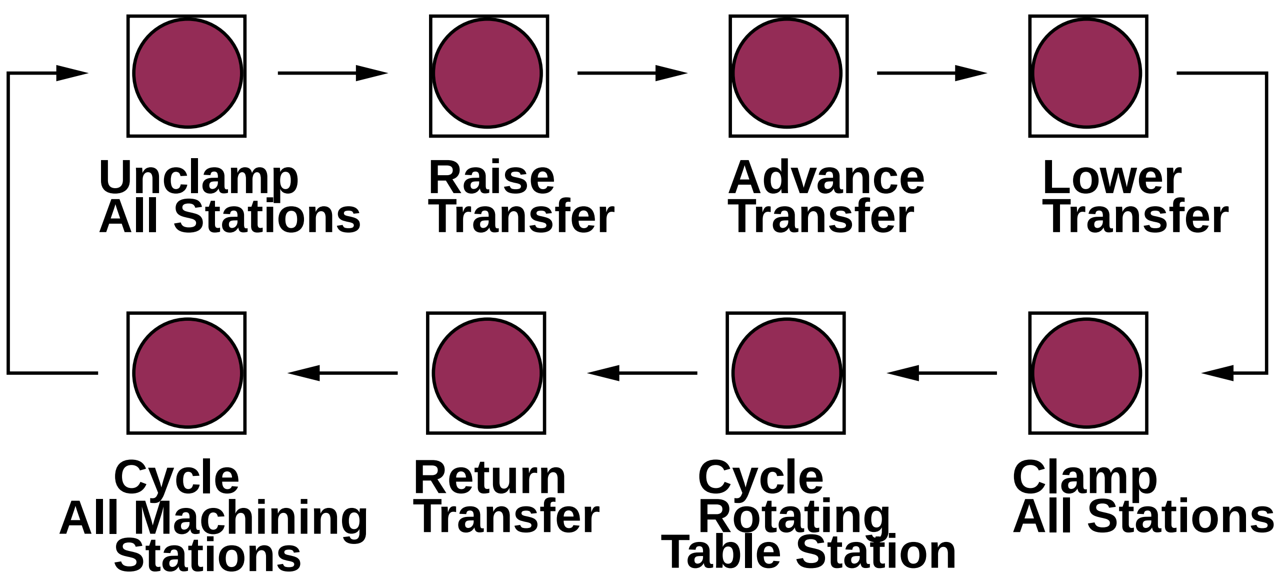


Outline

- Motivation: Logic control problem for high-volume machining systems
- Background
- Automatic cycle logic control
- Multi-mode logic control
  - What happens when things go wrong?
- Implementation and future work

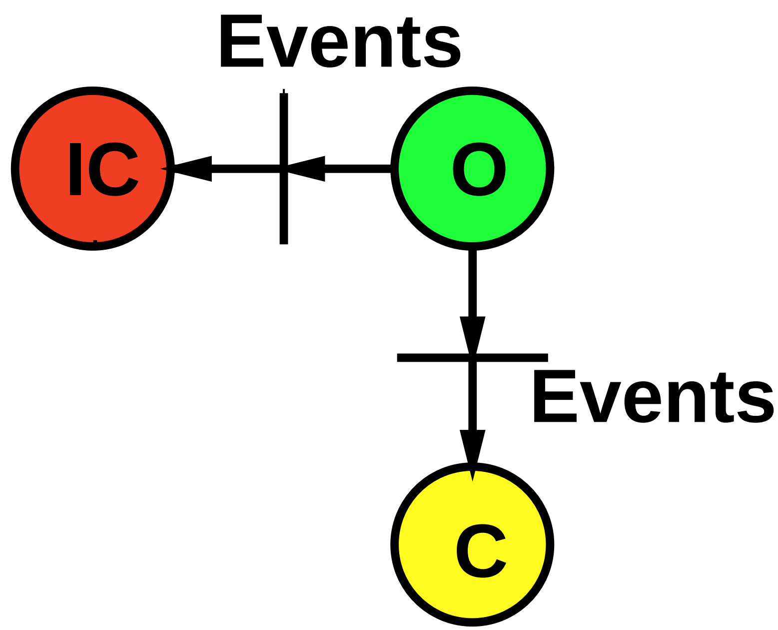
Operator interaction with control

- **Manual mode**
  - Single station fine operation control
- **Hand mode**
  - Normal operation cycle without overlapping
  - Reverse sequences possible



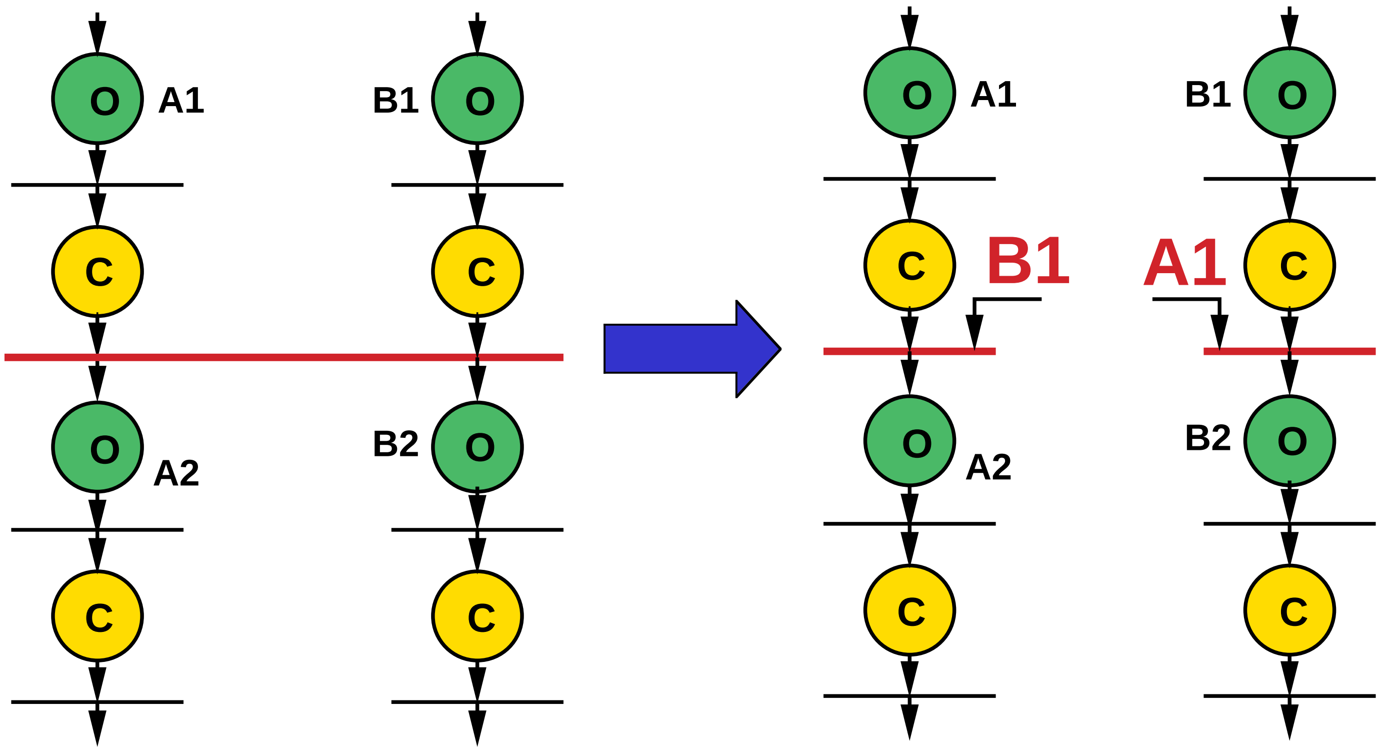
Three states for each operation

- Operating:
  - Activate actuator
- Completed stop:
  - Deactivate actuator
  - Set internal variable to 1
- Incomplete stop:
  - Deactivate actuator
  - Announce fault
- Sensors trigger transitions between states
- Internal variable used for coordination with other stations



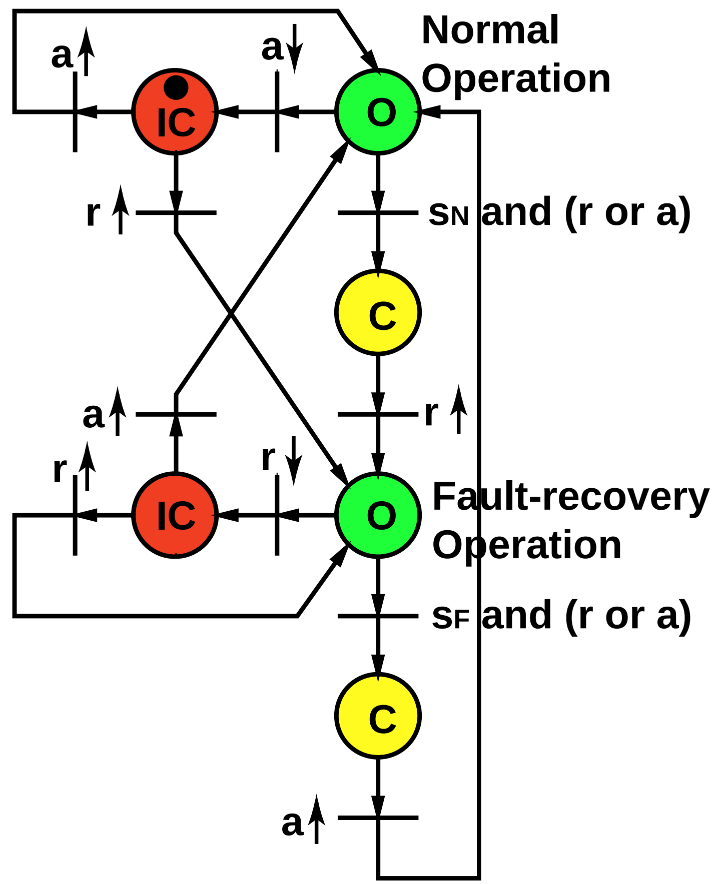
Internal variables to synchronize

- Synchronization conditions depend on control mode
- Decouple each control module



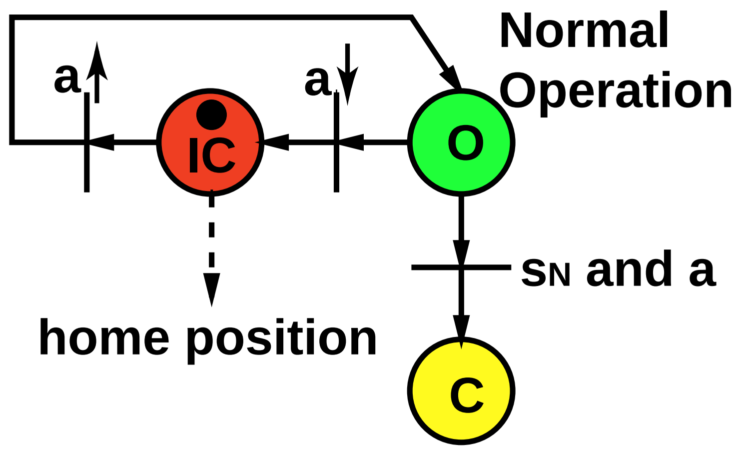
Reversible operation module

- **Normal and fault recovery operations**
- **Fault recovery:**
  - **Restart from fault position**
  - **Return to initial position of operation, then restart**
- **Faults may occur in fault recovery operation**
- **Transitions triggered by operator commands advance (a) and return (r) in manual mode**



Irreversible operation module

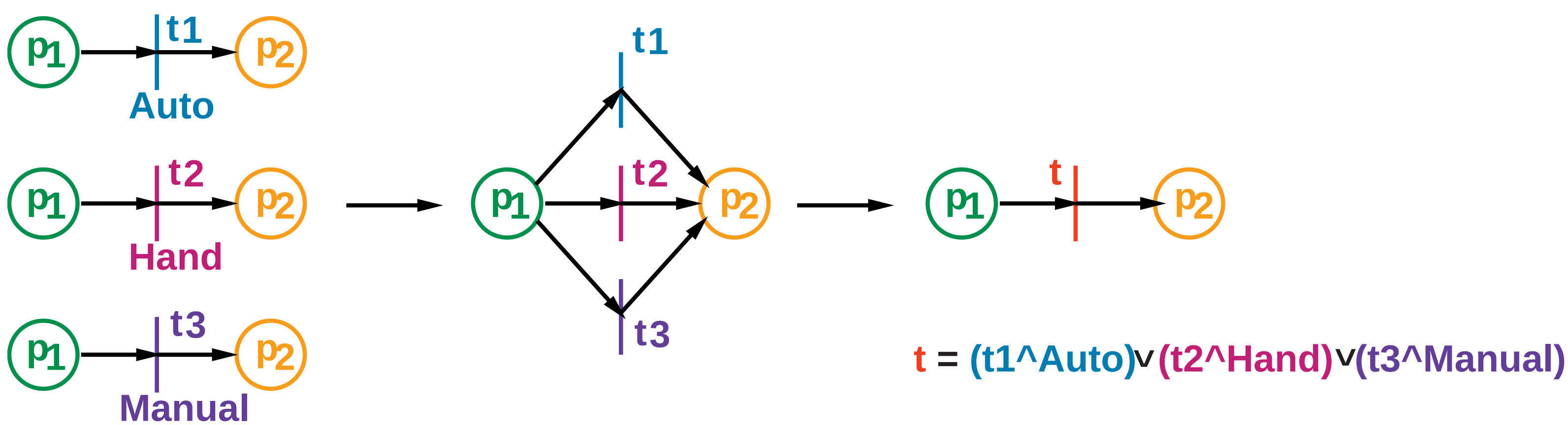
- Metal-removal operations (i.e. milling)
- Only contains normal operation block
- Fault recovery:
  - Restart from fault position
  - Return to home position of machining station and restart



- Transitions triggered by operator commands advance (a) and return (r) in manual mode

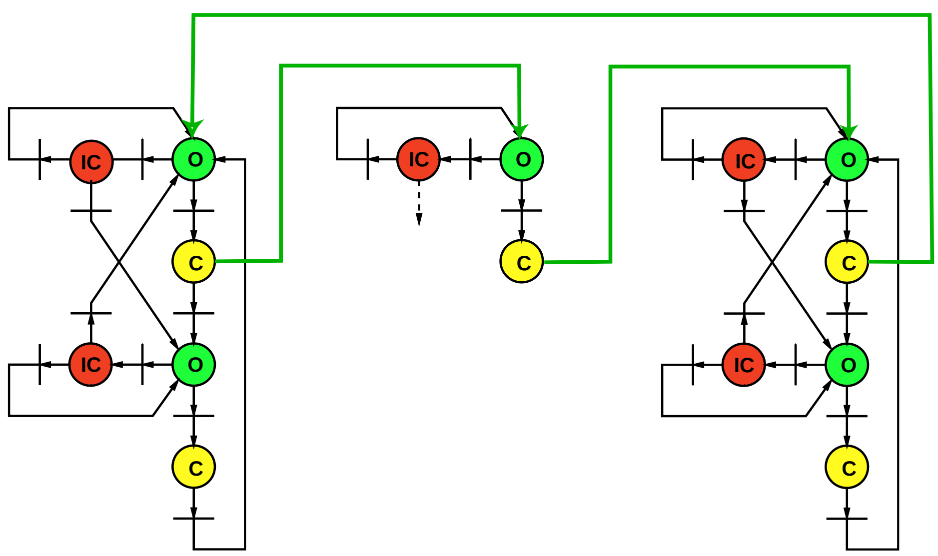
Combining control modes

- Same set of states for all modes
- Transition conditions depend on active control mode
- Superposition preserves liveness, safeness, reversibility properties



Algorithm to build logic controller

1. Assign module for each operation
2. Normal operation cycle ordered as in timing bar chart
   – Mill example: Advance, Feed, Return



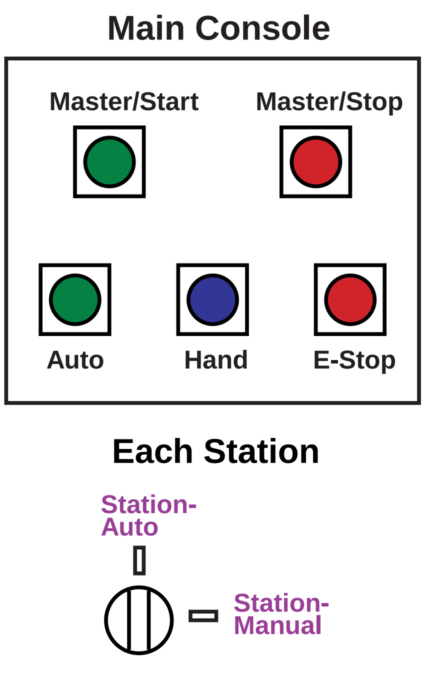
Algorithm to build logic control

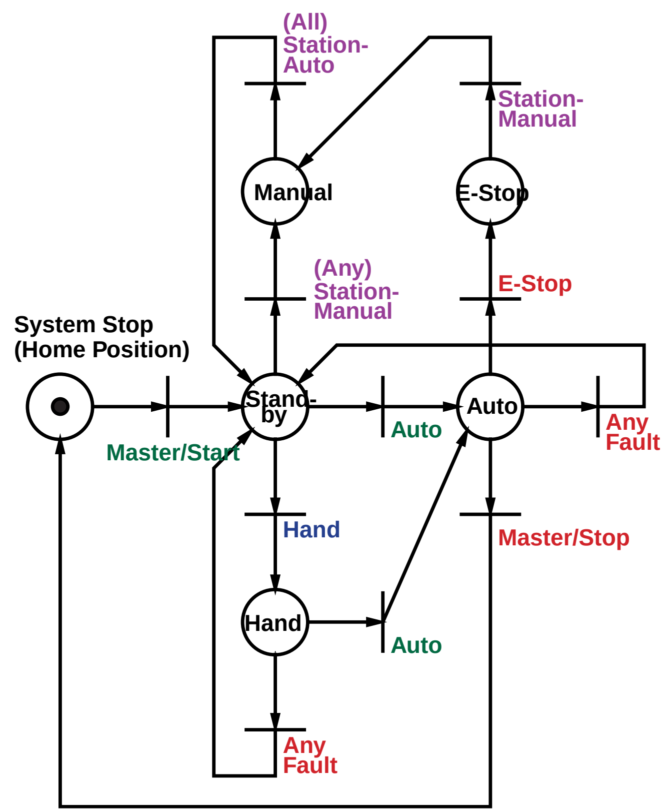
3. Reverse sequences for repeatable steps in hand mode
4. Fault recovery sequences for irreversible operations

Theorem: Station control module constructed according to algorithm is live, safe, and reversible Petri net.

Mode decision control logic

- Mode chosen by operator from input panel
- Mode decision control logic is live, safe, reversible





Modular logic controller

- **Operation causality condition to ensure well-ordered set of operations**
- **Station logic controllers with mode decision control logic**

Theorem: Resulting controller is guaranteed to be live, safe, reversible

Outline

- **Motivation: Logic control problem for high-volume machining systems**
- **Background**
- **Automatic cycle logic control**
- **Multi-mode logic control**
- **Implementation and future work**
  - **To the factory floor**

Implementation in PLC

Sequential function charts (SFC)

– IEC 1131-3 standard language

– Based on Petri nets

– One-one translation

Marked Graph	SFC/Grafcet
Simple Place	Simple Step
Initial Place	Initial Step
Simple Transition	Simple Transition
Synchronized Transition	Synchronized Transition
Macro Place	Macro Step

Industrial implementation

- US Patent applied for, 1998
- Current cooperation with Lamb Technicon on Cummins Engine project
- Evaluating implementation needs
  - Machine services
  - Safety and gate interlocks
  - Reusability
  - PLC platform dependence
  - User interface
- Commercialization potential

Future of reconfigurable control

- Unified framework for continuous and discrete control for machining systems
  - Modular structures for reconfigurability
  - Mathematical basis for verification
  - Integrated diagnostics
  - Automatic fault detection and recovery
- Software tools for control design/analysis
  - Interface with mechanical design software
  - Automatic control code generation