Application of Formal Methods for Designing PLCs

A Petri Net Approach

Ali Pouyan
Email: apouyan@ieee.org

Programmable Logic Controller -PLC

- A PLC is a specifically designed digital control device that accepts input signals and emits output signals based on a control logic program.
- PLCs are commonly used as sequence controllers in industrial automated systems.

Characteristics of a PLC

- It has a programmable memory to store the instructions.
- The stored instructions in the memory are used to implement several functions such as: logic, sequencing, timing, counting, and arithmetic functions.
- The various functions are used to control machines or processes.

PLC - History

- The history of Programmable Logic Controllers (PLCs) goes back only as far as 1968 to the General Motors Corporation (GM).
- Prior to 1968, the industrial control functions were performed by control relays (CRs).

CRs Disadvantages

- They are capable only of on/off control, so the control scheme for complicated systems is quite expensive.
- They are quite bulky.
- They are power-hungry, so high power consumption results in heat generation.
- When a relay fails, troubleshooting or locating the failed relay is difficult.
- They are hardwired, so any change in the control program requires relays to be rewired.

GM Criteria for PLCs -1968

- Easily programmed or reprogrammed with a minimum of downtime.
- Easily maintained.
- Rugged enough to operate in an industrial environment.
- Not power-hungry.
- Competitive in cost.

First Generation PLCs

- The major characteristic of the first generation was that they did have *primitive self-diagnostic indicators* that aided troubleshooting.
- As time progressed, PLCs acquired the capacity to do arithmetic, manipulate data, and communicate more efficiently with the programmer.

Today's PLCs

- They are economical, user-friendly devices that have benefited from advances in microprocessor and memory technologies.
- Today's PLCs are capable of performing complicated control routines and can communicate with other PLCs and host computers in sophisticated control networks.
- As time progressed, PLCs acquired the capacity to do arithmetic, manipulate data, and communicate more efficiently with the programmer.

Today's PLCs - continued

- It is widely believed that one of the areas where systems engineering methods still need a consistent development is that of logic controllers.
- Specification formalism and design criteria are still far from basic analytic requirements, in this area.

Statement of the Problem

- The problem of control leads to the design of sequence control software for automated discrete event dynamic systems (DEDS).
- Sequence control is mainly asynchronous, with concurrent control to maintain synchronisation and exclude conflicting machine actions.

Statement of the Problem

- The existing (modelling) languages are fundamentally procedural languages and it is still difficult to understand the program's behaviour when program size becomes large.
- They do not provide formal verification for the designed control software.

PLC Standard Languages - IEC

- IL: Instruction List
- ST: Structured Text
- FBD: Function Block Diagram
- SFC: Sequential Function Chart
- LLD: Ladder Logic Diagrams

None of the above languages are design methodologies, nor are they widely accepted or tested, except LLD.

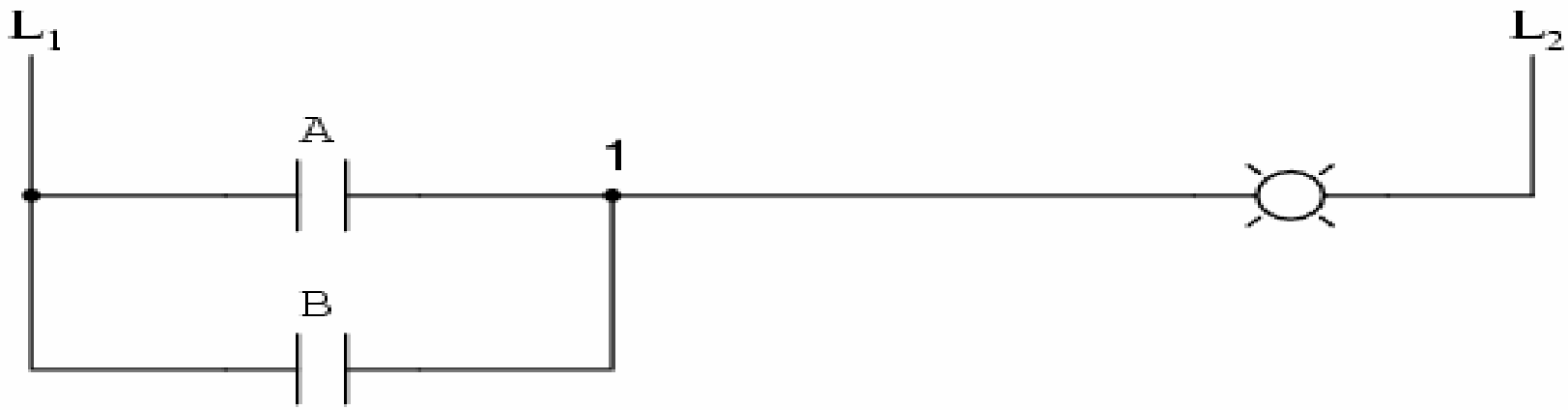
LLDs

- Even if LLDs assume a relevant importance in the industrial market, there is not yet a standard integrated tool, which is sufficiently simple to use, powerful, versatile, unambiguous, and with which it is possible to carry out proofs on formal correctness besides the classical validation through simulation.

Building Blocks of Logic Controllers

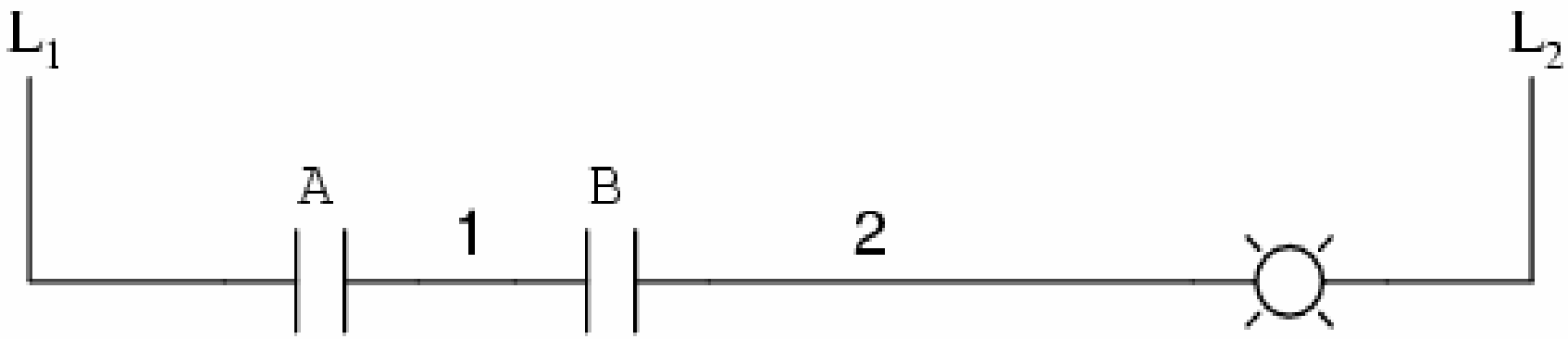
- Logical OR
- Logical And
- Concurrency
- Synchronization

Logical "OR"

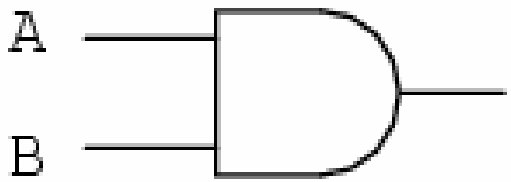


A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

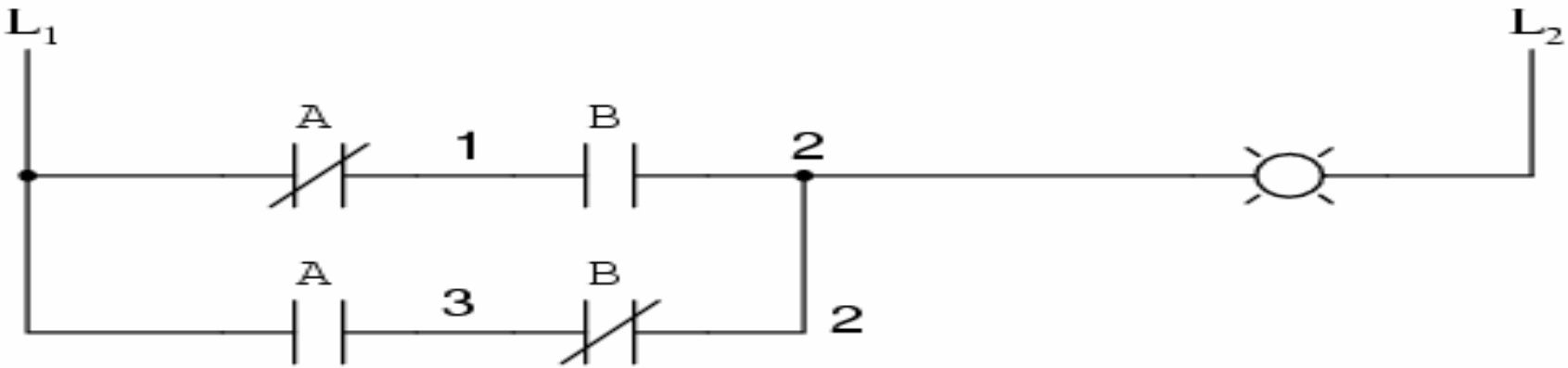
Logical "AND"



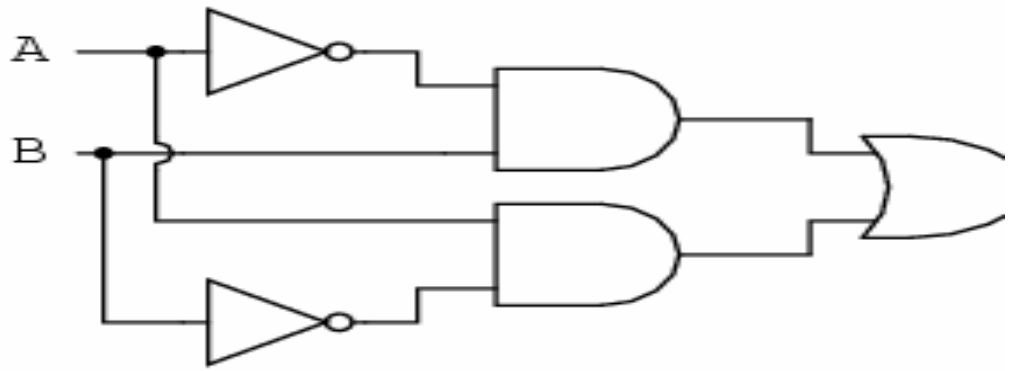
A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1



Combinational Logic Functions



A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0



or

Proposed Method

- We have proposed a conceptual and practical methodology to design control software systems (Logic controllers) with Petri Net theory.

Proposed Method

- In particular, the proposed specification formalism is based on a special class of Petri nets, tailored to model the control flow in discrete systems.

Proposed Method

- The main objective of the proposed methodology is to reduce the design effort due to the complexity of control problems and to provide design criteria that fit the fundamental engineering approach to complexity: the incremental design.

Proposed Method

- The methodology is based on concepts coming from a number of proven theorems.

Petri Nets: A Snapshot

- In 1962 Carl Adam Petri, a German mathematician, introduced Petri nets as a mathematical tool for describing relations between conditions and events.
- Now Petri net theory is an important field in theoretical computer science as a high level formal specification language.

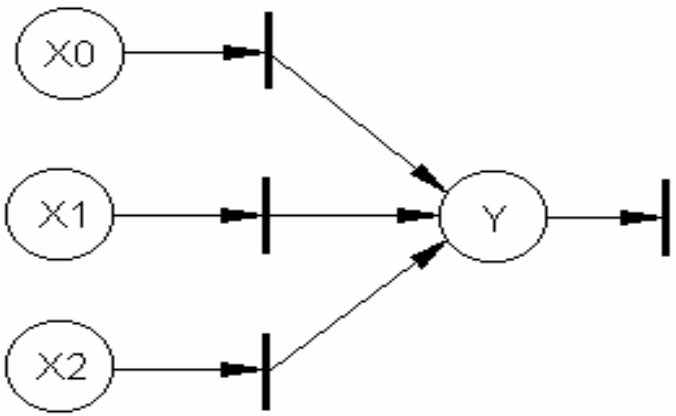
Petri Nets: A Snapshot

- Petri nets can be shown graphically using a 3-element set of symbols:
- a place: represented by circle
- a transition: represented by a bar or a box
- an arc: represented as a directed arc connecting either a place to a transition or a transition to a place

Petri Nets: A Snapshot

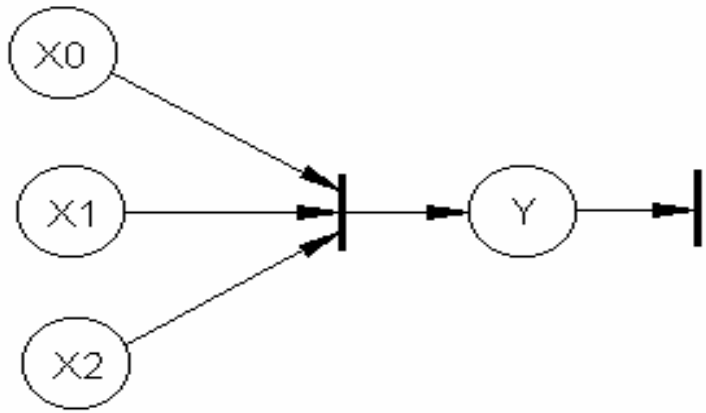
- A Petri net structure, N, is defined as a four-tuple, N = (P, T, V, F) where
- $P \cup P$ = { p1,p2,..., pn } is a finite set of places, $n \geq 0$,
- T = { t1,t2,..., tm } is a finite set of transitions, $m \geq 0$, ($T \cup P$ form the nodes of N),
- $V \subseteq \{ (P \times T) \cup (T \times P) \}$ is a set of directed arcs (or a flow relation), and
- $F : V \rightarrow \aleph$ is a multiplicity function, $\aleph = \{0,1,2,\ldots\}$, $P \cap T = \varnothing$ and $T \neq \varnothing$.

Building Blocks of the Sequence Control



IF X0 = 1 OR X1 = 1 OR X2 = 1
THEN Y =1.

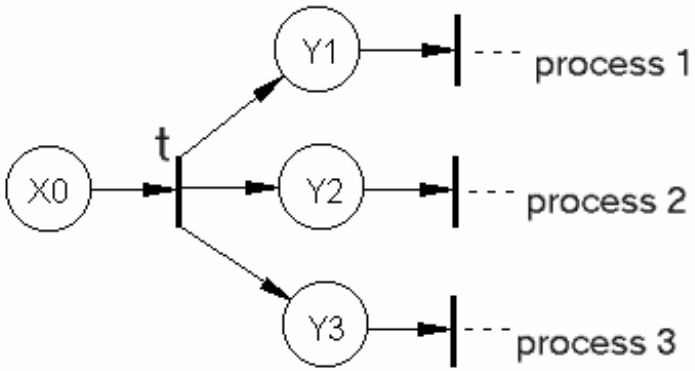
(a) Logical OR



IF X0 = 1 AND X1 = 1 AND X2 = 1
THEN Y =1.

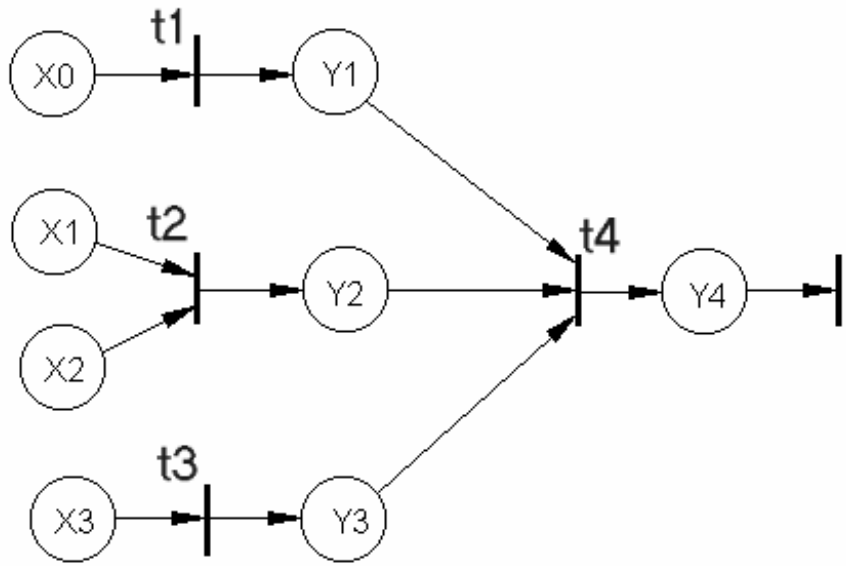
(b) Logical AND

Building Blocks of the Sequence Control



IF X0 = 1
THEN Y1 = 1 and Y2 = 1 and Y3 = 1

(a) Concurrency



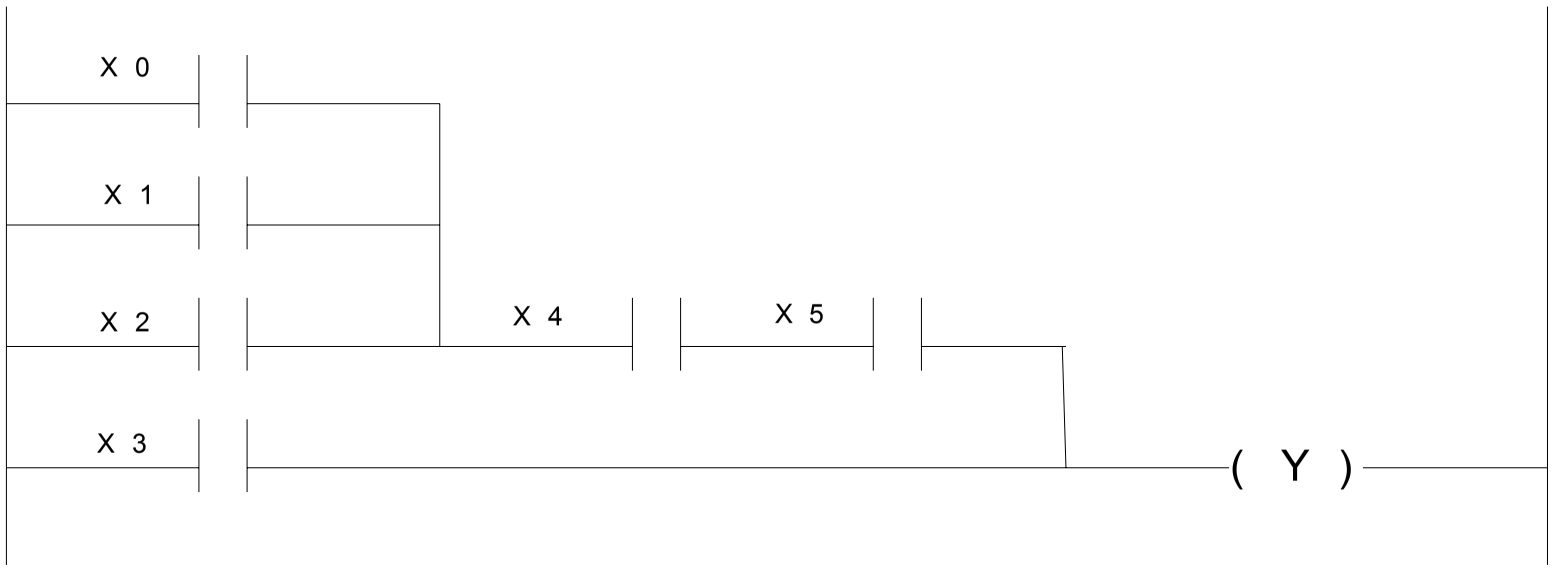
IF X0 = 1 THEN Y1 =1,
IF X1 =1 AND X2 =1 THEN Y2 =1,
IF X3 = 1 THEN Y3 =1,
IF Y1 = 1 AND Y2 = 1 AND Y3 = 1
THEN Y4 =1.

(b) Synchronisation

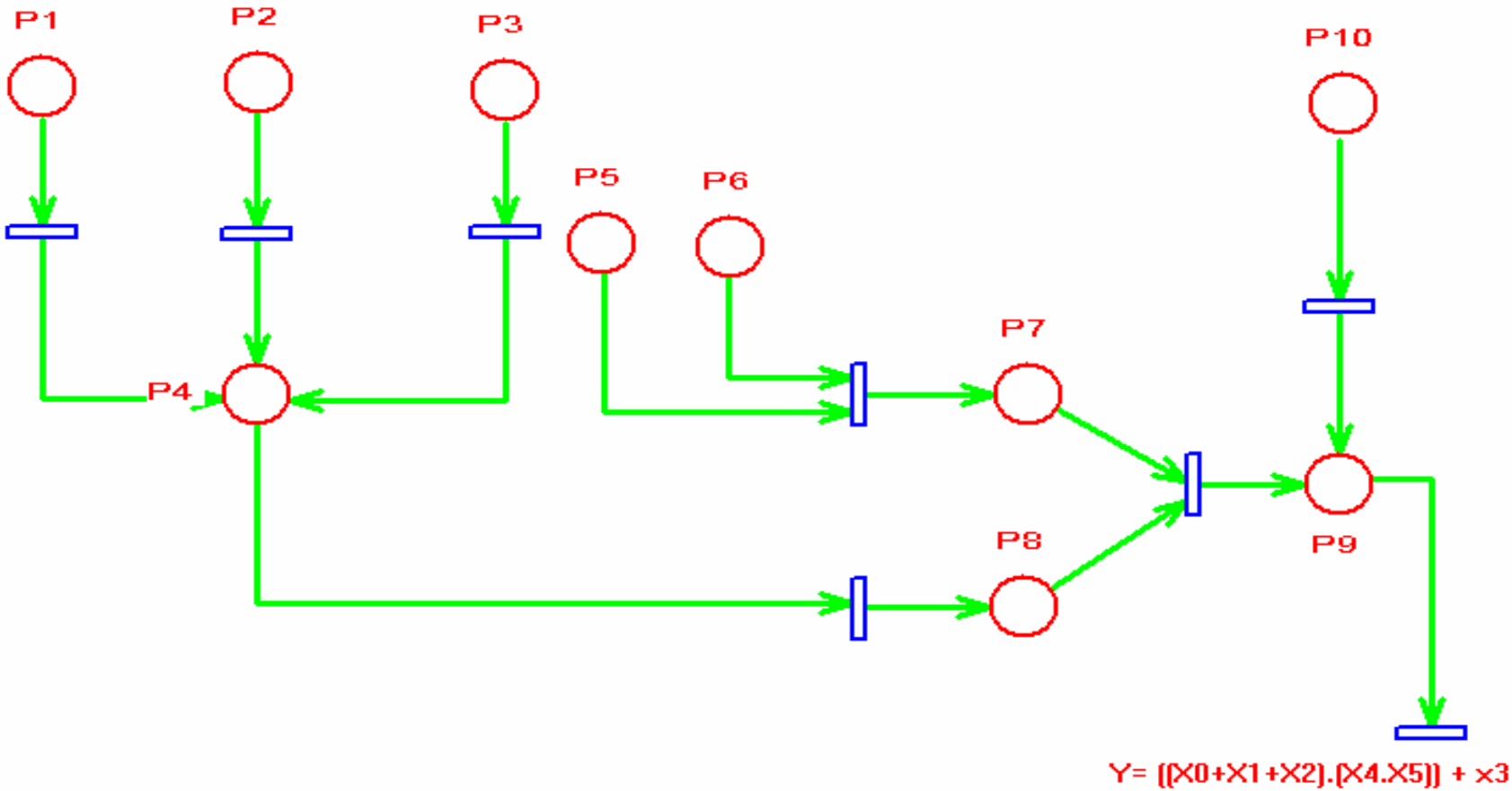
LLD of a Logic Equation

$$Y = ((X0+X1+X2).(X4.X5)) + X3$$

(b)



Petri Net Model of a Logic Equation



Disadvantages

- LLD-based controllers become cumbersome both to design and debug when applied to complex sequential tasks in discrete-event manufacturing systems.
- When a problem is detected in the system, cause-tracing and locating becomes extremely difficult.
- LLDs don’t provide any tools for analysis and performance characteristics of the system other than basic simulation for verification of programs.

Disadvantages

- In the context of agile manufacturing systems, in which control sequences need to be regularly modified, to meet the dynamically changing requirements of the outer environment, LLD-based sequence controllers are not readily adaptable to the changes needed to fulfil the system specification change.

Petri Net Model vs. LLD Model

- Both theoretically and practically, a formal (PN) model of a circuit that can be analysed and verified is always preferred to a LLD static graph.
- The dynamics of a system can be studied through the transition firings and distribution of tokens in a PN model.

Characteritics of the Proposed Approach

- **There is no need to construct separate models (modules) of different control tasks and merge them to build up an expanded system. The system is expanded by knitting the control tasks to the basic process in a structural rule-based incremental fashion while preserving the desired properties.**

Characteristics of the Proposed Approach

- **The logic controller can be designed at any level of abstraction. Furthermore, a detailed expanded system can be reduced to a system with higher level of abstraction according to certain rules.**

Characteristics of the Proposed Approach

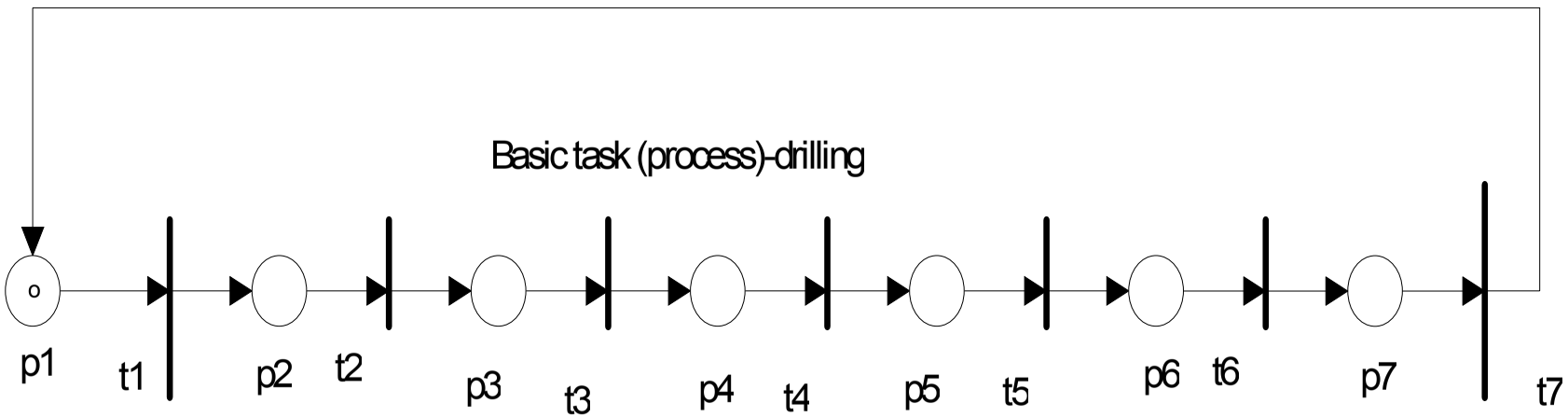
- **It is a rule-based approach which can be implemented (cast to code) as a CAD tool for logic controller design**

Example

- Consider a manufacturing cell consisting of a revolving bench with three workstations: loading, drilling, and quality control and unloading. The revolving bench is rotated to put the pieces under correct machines. Three processes can progress concurrently, ie., loading, drilling, and quality control.

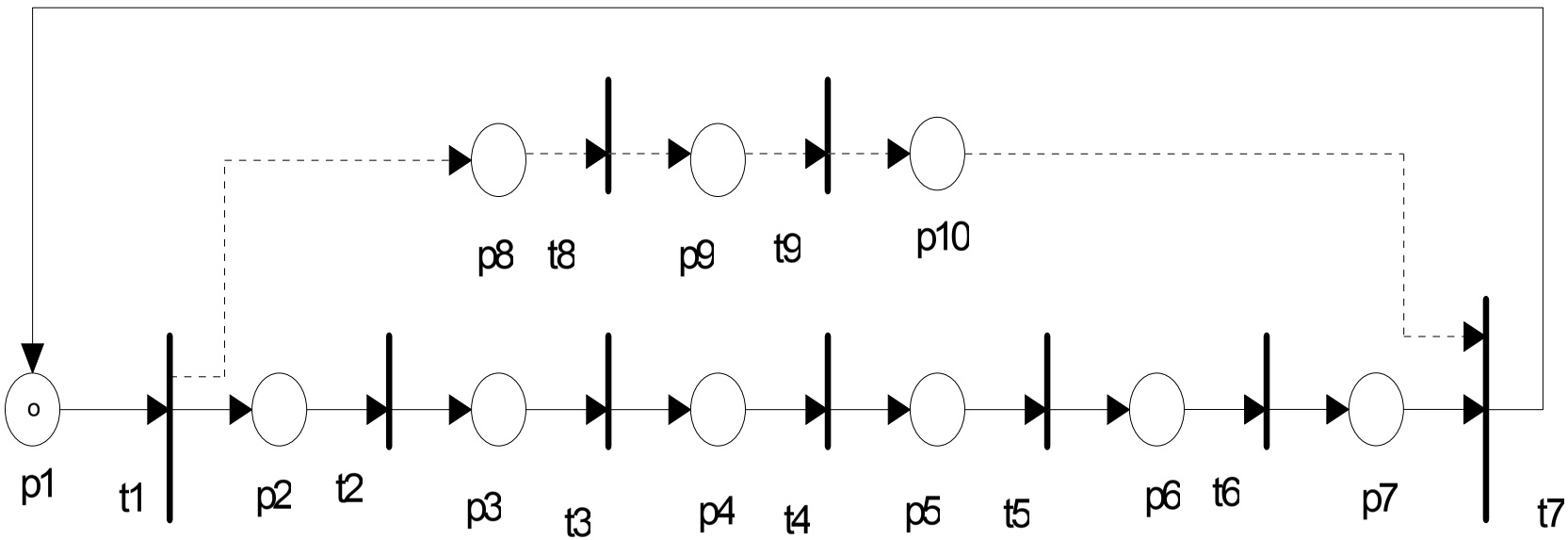
Example -1

The drilling task modelled by a Petri Net



Example -2

Integrating the loading task to the system



Example - 3

The final model after integrating quality control task