

Fall 2010

Modeling next generation air traffic control system with petri net

Hang Wu

New Jersey Institute of Technology

Follow this and additional works at: <https://digitalcommons.njit.edu/theses>



Part of the [Computer Engineering Commons](#)

Recommended Citation

Wu, Hang, "Modeling next generation air traffic control system with petri net" (2010). *Theses*. 85.
<https://digitalcommons.njit.edu/theses/85>

This Thesis is brought to you for free and open access by the Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Theses by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

ABSTRACT

MODELING NEXT GENERATION AIR TRAFFIC CONTROL SYSTEM WITH PETRI NET

**by
Hang Wu**

The Federal Aviation Administration (FAA) is one of the largest Air Navigation Service Providers, managing air traffic for more than 15% of the world's airspace. Today's Air Traffic Control (ATC) system cannot meet the growth of the air traffic activities, which brings with more unprecedented delays. At the same time, Air Traffic Controllers are facing higher workload than ever before. The FAA has declared that the existing ATC system will transition to a new system known as "Free Flight". "Free Flight" will change today's ATC system by giving pilots increased flexibility to choose and modify their routes in real time, thereby reducing cost and increasing system capacity (Nordwall, 1995). In this work, the modules and data flow of the next generation ATC system is designed, and their Petri net models are constructed for the control module to achieve "Free Flight".

**MODELING NEXT GENERATION AIR TRAFFIC CONTROL SYSTEM
WITH PETRI NET**

**by
Hang Wu**

**A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Engineering**

Department of Electrical and Computer Engineering

January 2011

Blank Page

APPROVAL PAGE

**MODELING NEXT GENERATION AIR TRAFFIC CONTROL SYSTEM
WITH PETRI NET**

Hang Wu

Dr. Mengchu Zhou, Thesis Advisor Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Edwin Hou, Committee Member Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Hesuan Hu, Committee Member Date
Adjunct Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Hang Wu

Degree: Master of Science

Date: January 2011

Undergraduate and Graduate Education:

- Master of Science in Computer Engineering,
New Jersey Institute of Technology, Newark, NJ, 2011
- Bachelor of Science in Computer Engineering,
New Jersey Institute of Technology, Newark, NJ, 2009
- Associate Degree in Computer Science,
Tong Ji University, Shanghai, P. R. China, 1999

Major: Computer Engineering

To My Loved One for All the Support and Encouragement.

ACKNOWLEDGMENT

This Thesis owes its existence to the help, support, and inspiration of many people. First of all, I would like to express my deepest sense of gratitude to my advisor Dr. Mengchu Zhou for his patient guidance, encouragement and excellent advice throughout this study. Without his help, this work would not be possible. I would also like to express my gratitude to the members of my Thesis Committee Dr. Edwin Hou and Dr. Hesuan Hu. Their advice and patience are appreciated. I would like to thank Dr. Hesuan Hu for his generous assistance during this time. Special thanks also to Ms. Gonzalez-Lenahan and Ms. Jenita Desai for their comments and suggestions for editing my Thesis writing.

Finally, I take this opportunity to express my profound gratitude to my beloved wife, Ling Han for her continuous and unconditional support and patience during my study at NJIT.

TABLE OF CONTENTS

Chapter	Page
1 INTRODUCTION.....	1
2 DESCRIPTION OF CURRENT AIR TRAFFIC CONTROL SYSTEM	3
3 PETRI NET	9
4 OVERVIEW OF NEXT GENERATION ATC SYSTEMS	12
5 PETRI NET MODELS OF ATC SYSTEM	17
5.1 Introduction	17
5.2 Departure and Arrival Subnets	18
5.3 Combined Models	20
5.3.1 A Departure Model	20
5.3.2 An Arrival Model	22
5.3.3 One-runway One-direction Petri Net Model	23
6 SIMULATING AND ANALYSIS	34
6.1 Simulating via VHDL	34
6.2 Analysis via INA	40

TABLE OF CONTENTS
(Continued)

Chapter	Page
7 CONCLUSION	45
APPENDIX A VHDL CODE FOR AIRPORT TWOER ATC CONTROL UNIT.....	47
APPENDIX B INA SOURCE CODE FOR AN ARRIVAL PETRI NET MODEL.....	51
APPENDIX C INA SOURCE CODE FOR A DEPARTURE PETRI NET MODEL.....	52
APPENDIX D INA SOURCE CODE FOR ONE-DIRECTION PETRI NET MODEL.....	54
APPENDIX E INA SOURCE CODE FOR EN ROUTE PETRI NET MODEL.....	55
REFERENCES	57

LIST OF TABLES

Table		Page
1.1	The Ratio of Flight Operations for 2030 vs. 2010.....	1
5.1	Description of en route Petri net Control Model.....	31
6.1	Description of Input and Output Signals for a Control Unit.....	35

LIST OF FIGURES

Figure	Page
2.1 United States Air Space	4
4.1 WAAS System	13
4.2 Las Vegas Air Space	14
4.3 Next generation ATC System Overview	15
5.1 A FACET snapshot of air traffic over the United States.....	18
5.2 Arrival flight paths of Hong Kong International Airport.....	19
5.3 Departure flight paths of Hong Kong International Airport.....	19
5.4 Petri net representation of a departure model.....	20
5.5 Petri net representation of an arrival model.....	22
5.6 One-runway One-direction Petri net model.....	24
5.7 Petri net model for Hong Kong International Airport ATC system.....	26
5.8 2-D view for horizontal operations.....	27
5.9 2-D view for vertical operations.....	28
5.10 Petri net model for en route control.....	30
6.1 Simulation waveform for handling arrival flight.....	36

LIST OF FIGURES
(Continued)

Figure	Page
6.2 Simulation waveform for handling departure flights	37
6.3 Simulation waveform for handling bypass case due to emergency	38
6.4 Simulation waveform for handling departure and arrival flights simultaneously	39
6.5 INA analysis result for an arrival model	41
6.6 INA analysis result for a departure model	42
6.7 INA analysis result for one-direction model	43
6.8 INA analysis result for en route model	44

CHAPTER 1
INTRODUCTION

The safe, reliable and efficient management of our ever-increasing air traffic is one of the fundamental challenges today. On a typical day, over 40,000 commercial flights operate within US air space (Sridhar & Dohi & Sherth & Chatterji, 2006). The volume of air traffic is increasing at least as fast as the general economy. Today's centralized control systems cannot meet the rapid growth of the air traffic. The FAA compares the ATC forecast activity levels for 2030 with the report's estimates for 2010. Table 1 provides the relevant activity levels, by type of users, for towers, TRACONs and Centers, where TRACON stands for Terminal Radar Approach Control. It shows the ratios of flight operations for 2030 vs. 2010. There will be significant growth in ATC activity over the next 20 years.

Table 1.1 Ratios of Flight Operations for 2030 vs. 2010

	Airlines	Air Taxi/Commuter	Gen. Aviation	Military	Total
Tower Ops.	1.56	1.34	1.29	1.0	1.35
TRACON Ops.	1.65	1.32	1.27	1.0	1.40
Center IFR Aircraft	1.94	1.34	1.20	1.0	1.63

Source: Poole, Robert Air Traffic Control Reform (2010).
From <http://reason.org/news/show/air-traffic-control-march-2010> accessed October 9, 2010.

Tower operations in 2030 will be 35% higher than 2000, TRACON operations 40% higher, and Center operations 63% higher. Airline operations in the system will grow by 50 to 90%, air taxi by 30%, while general aviation by 20 to 30% and military will be essentially at the same level.

Today's ATC system is a centralized control system. Air traffic controllers monitor and control the whole air traffic system. They work within The National Airspace System (NAS) to coordinate the movement of air traffic. The growth rate of ATC activities is much higher than the growth of the capacity of the current ATC system. Increased air traffic activities lead to increase in the workload of air traffic controllers, which will cause congestion and resulting delays.

In the near-term, FAA may suggest focusing implementation efforts on the tower control that represents the largest bottlenecks in the system. On an aggregate basis, however, the largest growth will be in the en-route portion of the system. In that case, "free flight" should be on implementation there to increase system capacity.

The next generation ATC system must achieve a large increase in capacity and throughput while improving efficiency and safety. This paper presents a Petri net approach to modeling the control unit of next generation ATC systems to achieve "free flight".

The rest of the thesis is organized as follows; Chapter 2 describes the current air traffic control system. Chapter 3 discusses the basic definitions of Petri nets. Chapter 4 overviews the next generation ATC system. Chapter 5 discusses the Petri net models of the ATC system. Chapter 6 discusses simulation and analysis of Petri net models. Chapter 7 concludes the thesis.

CHAPTER 2

DESCRIPTION OF CURRENT AIR TRAFFIC CONTROL SYSTEM

An ATC system can be viewed as a vast network of people and equipment that ensure the safe operation of commercial and private aircraft. Air traffic controllers coordinate the movement of air traffic to make sure that planes stay at a safe distance apart. Their immediate concern is safety, but controllers must also direct planes efficiently to minimize delays (Transportation Research Board, 2003).

Twenty-four hours a day, seven days a week, controllers are on the job separating aircraft at over 350 locations across the United States. On any given day, more than 87,000 flights are in the skies in the country. On an average day, air traffic controllers handle 28,537 commercial flights (major and regional airlines), 27,178 general aviation flights (private planes), 24,548 air taxi flights (planes for hire), 5,260 military flights and 2,148 air cargo flights (Federal Express, UPS, etc.). At any given moment, roughly 5,000 planes are in the skies above the United States. In one year, controllers handle an average of 64 million takeoffs and landings. An ATC system is a service provided by ground-based controllers who direct aircraft on the ground and in the air. The primary purpose of ATC systems worldwide is to separate aircraft to prevent collisions, to organize and expedite the flow of traffic, and to provide information and other support for pilots.



Figure 2.1 United States air space.

Source: Freudenrich, Craig How Air Traffic Control Works (2007).

From <http://science.howstuffworks.com/transport/flight/modern/air-traffic-control.htm> accessed October 9, 2010.

United States airspace is divided into 21 zones (**centers**), and each zone is divided into sectors. Also within each zone are portions of airspace, about 50 miles in diameter, called **TRACON** airspaces. Within each TRACON air space are a number of airports, each of which has its own airspace with a 6-mile to 9-mile radius.

An ATC system can be divided into several subsystems –Air Route Traffic Control Centers (ARTCC), TRACON, Air Traffic Control Tower (ATCT), and Flight Service Station (FSS), based on the airspace division. Every aircraft that flies follows a

similar flight pattern that begins before takeoff and ends after landing. This pattern is called a flight profile. A typical commercial flight profile has seven phases. Each phase of a typical flight profile is monitored by an ATC facility with its own group of controllers. Each of these controllers follows specific rules and procedures while directing flights through designated airways. They monitor the flight using special equipment and decision support tools (computers) that ensure a safe and efficient flight.

The aircraft, pilot and air traffic controllers interact during each phase of a typical commercial flight profile. Their interactions are described as follows (NASA Air Traffic Management, 2003).

PREFLIGHT:

The pilot receives the most recent weather information and files a flight plan with air traffic control, prior to takeoff, the pilot performs the flight check routine, pushes back the aircraft from the terminal's gate, and is cleared by controllers to taxi out to the designated takeoff runways.

TAKEOFF:

The pilot receives permission from Local Control (the Tower) to takeoff. The aircraft powers up and begins its takeoff roll.

DEPARTURE:

Upon lift off, the pilot is instructed to change radio frequencies to receive new flight instructions from Departure Control in the TRACON. The pilot is instructed to follow a pre-determined, preferred routing that will take the aircraft up and away from the departure airport onto its route. The pilot is then issued further altitude and routing clearance. The controller monitors the target (the aircraft) and its track (flight path) on the

radarscope. As the aircraft reaches the edge of the TRACON airspace, the Departure Controller performs an electronic transfer of the flight to the controller in the next airspace.

EN ROUTE:

The pilot receives instructions as to what altitude and heading to maintain, as well as to which radio frequency to tune. This portion of the flight can be as short as a few minutes or as long as many hours.

DESCENT:

As the aircraft nears its destination airport, the pilot is instructed to change radio frequencies and contact Approach Control for instructions. The pilot is instructed to descend and change heading. After receiving these instructions, the aircraft descends and maneuvers to the destination airport.

APPROACH:

The pilot has received an approach clearance to the destination airport from the Approach Controller working in the TRACON. The flight has been placed in line with other aircraft preparing to land at the same airport. The pilot flies a specified flight procedure in order to get in line for the designated landing runway. The pilot receives instructions from the Approach Controller to change radio frequency and contact Local Control (in the airport's control tower) for landing clearance. The aircraft is electronically handed off from TRACON to the Tower.

LANDING:

The pilot receives clearance from the Local Controller in the airport's control tower to land on a designated runway. Upon touching down, the flight is then handed off to

Ground Control. The Ground Controller directs the pilot across the taxiways to its destination gate at the terminal.

Today's ATC system is a centralized control system. The Air traffic controllers are central decision makers and control the whole air traffic activities. Before takeoff, air traffic controllers approve the aircraft flight plans which cover the entire flight. During the flight, air traffic controllers send additional instructions to aircraft depending on the actual traffic, in order to improve traffic flow and avoid dangerous encounters until arriving the gate of its destination airport.

Ensuring the safety of aircraft is a controller's main priority, but another part of FAA mission is to guarantee the efficient flow of traffic through the NAS. Provided that safety is not compromised, airline companies, pilots, and the traveling public have an interest in efficient traffic flow. Controllers must address the sometimes-conflicting goals of safety and efficiency through an intricate series of procedures, judgments, plans, decisions, communications, and coordinated activities, in an environment in which errors may have dramatic consequences.

Air traffic controllers as decision makers working in complex environments make errors. In the context of ATC, Wickens *et al.* (1997) propose that there are two types of errors: operational errors and controller errors. An operational error is a formal designation and occurs when the reserved airspace of two aircraft overlap or when minimum separation criteria are not met between aircraft and terrain, obstacles, or obstructions (FAA, 1987). This type of error has more serious safety implications. Controller errors refer to a much wider range of inappropriate behaviors that result from

breakdowns in information processing (Diana & Giua, 2001). These errors may have minor safety implications or severe ones.

Most operational errors are made under conditions of moderate to light levels of workload, traffic complexity, and traffic volume, and when controllers are working under the combined radar/radar associate (Diana & Giua, 2001). Redding and his colleagues (1991) suggest that deficient Situation Awareness (SA) due to a lack of vigilance in monitoring cause many errors. Redding and Seamster (1994) confirm the previous findings when observing that most operational errors occur with traffic levels of moderate complexity, with an average of only eight aircraft under control, and immediately following a shift break. They also propose that failure to maintain adequate SA is a major cause of operational errors.

CHAPTER 3

GENERALITIES ON PETRI NET

In this section, the formalism is recalled and used in this work. Petri nets were conceived by Carl Petri as a mathematical means of describing activities, resources, and states of a system. They have been used to model, analyze and evaluate control system behavior. Petri nets have been widely used to model discrete –event dynamic systems e.g., manufacturing systems, transportation networks, computer networks and web services (Hruz & Zhou, 2007).

A marked Petri net $Z = (P, T, I, O, m)$, where

- (1) $P = \{p_1, p_2, \dots, p_n\}; n > 0$;
- (2) $T = \{t_1, t_2, \dots, t_s\}, s > 0$ with $P \cup T \neq \emptyset$ and $P \cap T = \emptyset$
- (3) $I: P \times T \rightarrow \{0, 1\}$;
- (4) $O: P \times T \rightarrow \{0, 1\}$;
- (5) $m: P \rightarrow \{0, 1, 2, \dots\}$.

In this definition, p_i is called a place, t_i a transition. I an input function defining the set of directed arcs from P to T , O an output function defining the set of directed arcs from T to P , and m is an n -dimensional marking whose i -th component represents the number of tokens in the i -th place p_i . Pictorially, places are represented by circles and transitions by bars. If $I(p, t)=1$, a directed arc is drawn from place p to transition t . If $O(p, t)=1$, a directed arc is drawn from t to p . A marking assigns to each place a nonnegative integer. If a marking assigns to place p a positive integer k , it is said that p is marked with k tokens. In that case, place k black dots (tokens) or number k (if k is big) in p .

The behavior of many systems can be described in terms of system states and their changes. In order to simulate the dynamic behavior of a system, a state or marking in a Petri net is changed according to the following transition enabling and firing rules:

- (1) A transition t is enabled if and only if $m(p) \geq I(p, t)$, $\forall p \in P$;
- (2) An enabled transition t fires at marking m' , yielding a new marking,

$$m(p) = m'(p) + O(p, t) - I(p, t), \quad \forall p \in P.$$

A marking m is said to be reachable from m' . Given Z and its initial marking m_0 , the reachability set is the set of all markings reachable from m_0 through various sequences of transition firings and is denoted by $R(Z, m_0)$. Reachability is a fundamental basis for studying the dynamic properties of any system.

A Petri net is said to be B -bounded or simply bounded if the number of tokens in each place does not exceed a finite number B for any marking reachable from m_0 . It is said to be safe if it is 1-bounded. For a bounded Petri net, from the initial marking m_0 , we can obtain all reachable markings.

A Petri net is said to be live if, no matter what marking has been reached from m_0 , it is possible to ultimately fire any transition of the net by progressing through some further firing sequence. The liveness implies that the system is free from deadlock.

A Petri net is said to be reversible if for each marking m in $R(Z, m_0)$, m_0 is reachable from it. Thus, in a reversible net one can always get back to the initial marking.

Some of the advantages of Petri nets as models for discrete event control include (Zhou and Dicesane, 1993), (Zhou and Venkatezh, 1998), (Li and Zhou, 2009), and (Wu and Zhou, 2010): graphical representation, solid foundations based on mathematics, the

existence of simulation and formal analysis techniques, and the existence of computer tools for simulation, analysis and control.

CHAPTER 4

OVERVIEW OF NEXT GENERATION ATC SYSTEM

In the current ATC system, air traffic controllers issue commands and pilots follow them. If pilots want to change their routes, they must make requests and receive an order from controllers. The controllers are responsible for safe operation. In contrast, Free Flight will let pilots change their routes with controller's intervention only when necessary to ensure adequate separation. In some definitions of Free Flight, pilots are responsible for avoiding conflicts. Aircraft fly direct routes to their destinations whenever possible, ignoring existing jet ways. However, they must avoid flying through restricted airspace, such as zones around military bases.

The next generation ATC system will be built on Global Positioning System (GPS) that provides reliable location and time information in all weather and at all times and anywhere on or near the Earth when and where there is an unobstructed line of sight to four or more GPS satellites. According to FAA, they are going to use the GPS in air traffic control systems by 2020, possibly as early as 2015. It requires that any aircraft flying within commercial airspace must have a GPS-equipped navigation system, which is able to beam information to ground control stations that will no longer have to rely exclusively on radar.

All aircraft are tracked with an augmented GPS system, such as WAAS that stands for the Wide Area Augmentation System, and broadcast their positions and intended routes to other aircraft and ground stations via a data link such as ADS-B system where ADS-B stands for the Automated Dependent Surveillance Broadcast.

WAAS is an extremely accurate navigation system developed for civil aviation. The ADS-B database keeps track of status of whole air space. It could be a centralized database or distributed databases stored in different control centers.

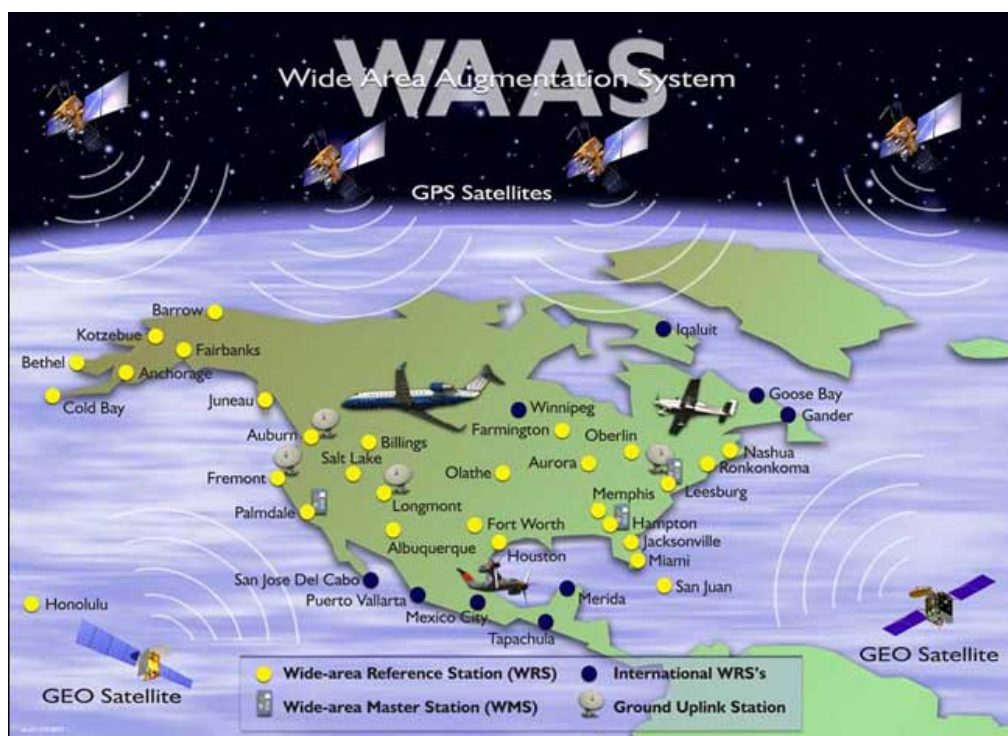


Figure 4.1 WAAS system.

Source: Navigation Services (2010). Wide Area Augmentation System (WAAS).
From <http://www.faa.gov> accessed October 9, 2010.

WAAS provides service for all classes of aircraft in all phases of flight - including en route navigation, airport departures, and airport arrivals. This includes vertically-guided landing approaches in instrument meteorological conditions at all qualified locations throughout the NAS (Navigation Service, 2010).

For example, Figure 4.2 represents the Las Vegas airspace. The air space showed in blue is available for aircraft to enter. On the other hand, the air space showed in red is unavailable. The reason could be bad weather happening there or it is a restricted

airspace. Las Vegas ADS-B database will store and update the status which air space controlled by Las Vegas ATC center. Any aircraft flying in the Las Vegas airspace can obtain airspace information from ADS-B database, in order to determine whether they can fly in certain air space or not.

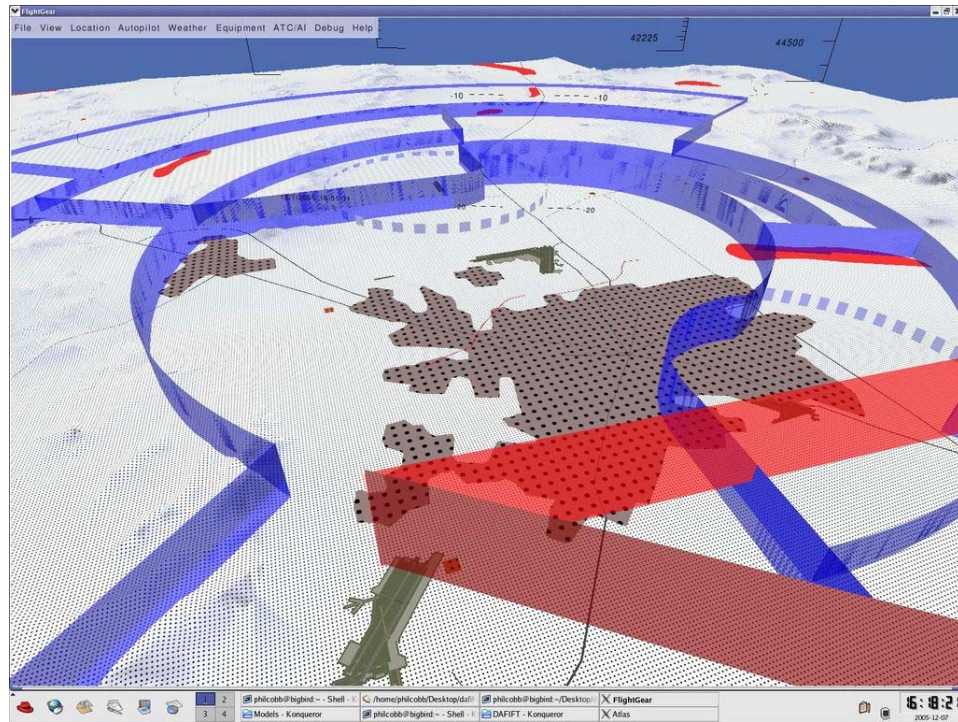


Figure 4.2 Las Vegas Air Space.

Source: Las Vegas Airspace (2005).

From <http://www.flightgear.org/Projects/SynthVision/Link/las-vegas-airspace.html> accessed October 9, 2010.

The next generation ATC system consists of several processes, which communicates with ADS-B database. Figure 4.3 shows the processes and communication flow.

The ADS-B database holds the whole airspace resource information and all aircrafts' flight data including altitude, latitude, speed, flight direction, etc. The control

and selection module requests air space resource information and flight data from the ADS-B system, and then calculates all available flight paths. It can make a decision by selecting a unit or processing the decision made by pilots. At the same time, it can send a selected flight path data to the flight path module.

A GUI module shows air space resource information and flight data from the ADS-B database to flight pilots and air traffic controllers. It also shows all available flight paths and all potential conflicts from the Control and Selection module to pilots or air traffic controllers. In that case, pilots and air traffic controllers can choose and send the selected solution back to the Control and Selection module.

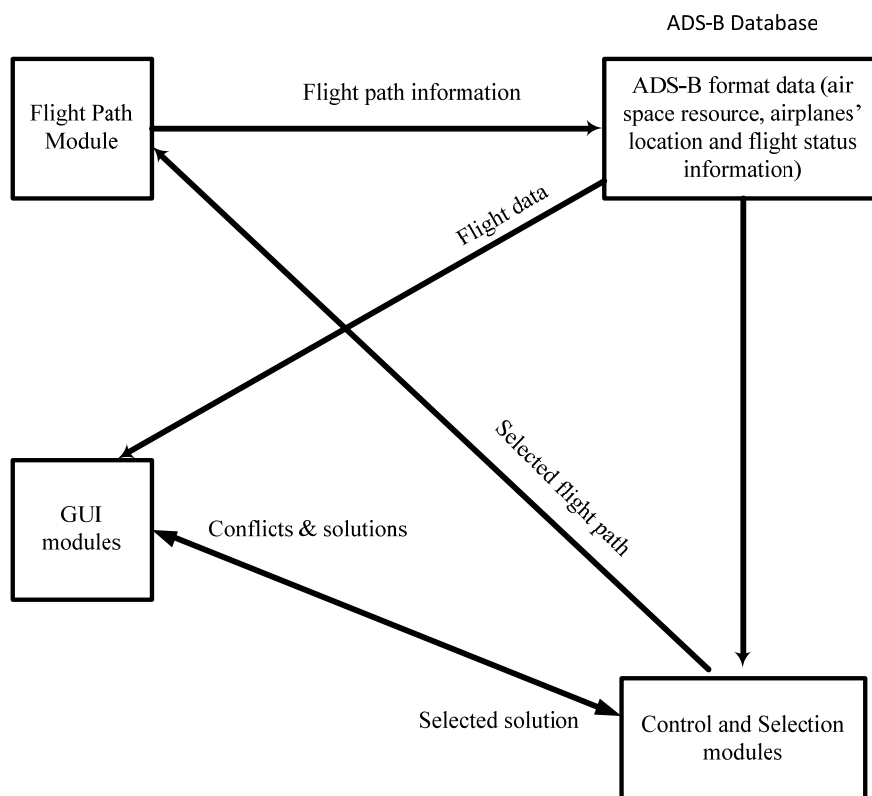


Figure 4.3 Next Generation ATC System Overview.

The flight path module implements the selected flight path data generated by the Control and Selection module and updates air space resource status and flight status in ADS-B database.

CHAPTER 5

PETRI NET MODELS OF ATC SYSTEM

5.1 Introduction

In this section, the Petri nets models of an ATC system is presented to handle the whole flight process, including departure, en route, approach, and landing.

Airports are end nodes in whole air space. In the ATC system, the airport towers control all aircraft flight in and out of airports. Figure 5.1 shows a FACET snapshot of air traffic over the United States on July 10, 2006. The air space around the airport has the highest flight density. In that case, the air traffic controllers who work in an airport tower have the highest workload. Airport air traffic controllers usually control several planes at a time. They have to make quick decisions about completely different activities. Aircraft have to be separated more than the minimum separation distance to avoid operational errors since the control signal is generated by human decisions under high workload. Most accidents or incidents happen in the air space near an airport. Airport towers become the bottle neck to improve the safety and efficiency of the whole air traffic system.

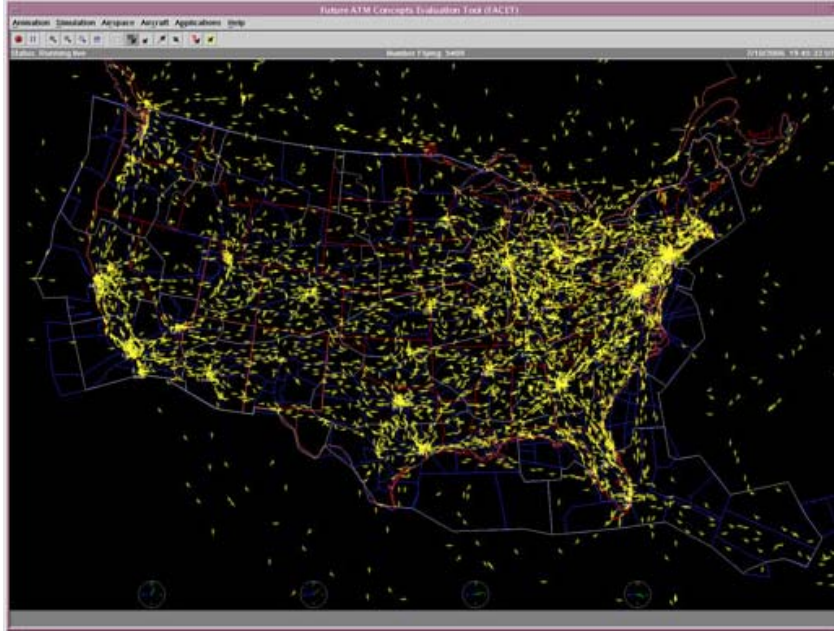


Figure 5.1 A FACET snapshot of air traffic over the United States on July 10, 2006 at 2:45 p.m. EST/ 11:45 a.m. PDT.

Source: Day In the Life of Air Traffic Over the United States (2006).

From <http://www.nasa.gov/vision/earth/improvingflight/FACETSOY.html> accessed October 9, 2010.

5.2 Departure and Arrival Subnets

Petri net model is created for airport tower control to generate a control signal to handle a flight's departure and arrival. The air space is three-Dimension. Aircraft can choose any flight routes with different directions. But aircrafts have to follow the same flight path to fly in and out of the airport. The flight path for arrival and departure is fixed once it is designed at every airport. For example, Figures 5.2 and 5.3 show the arrival and departure flight paths for Hong Kong international airport.

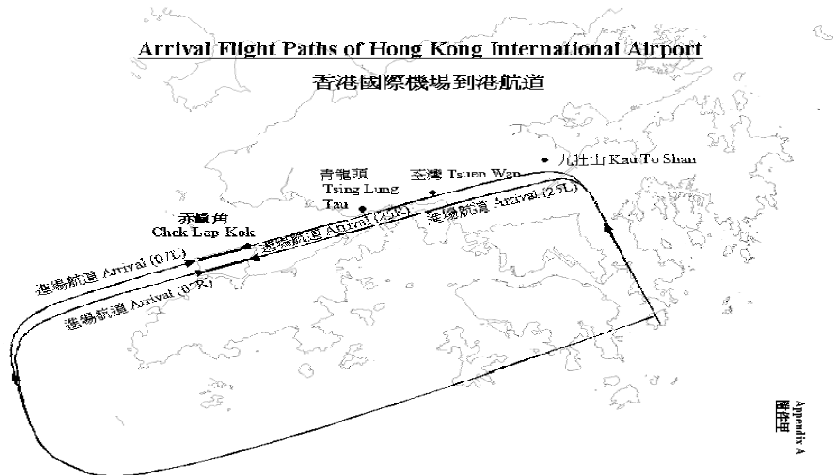


Figure 5.2 Arrival flight paths of Hong Kong international airport.

Source: Arrival Flight Paths of HK International Airport (2004).
 From http://www.cad.gov.hk/english/ac_path.html accessed October 9, 2010.

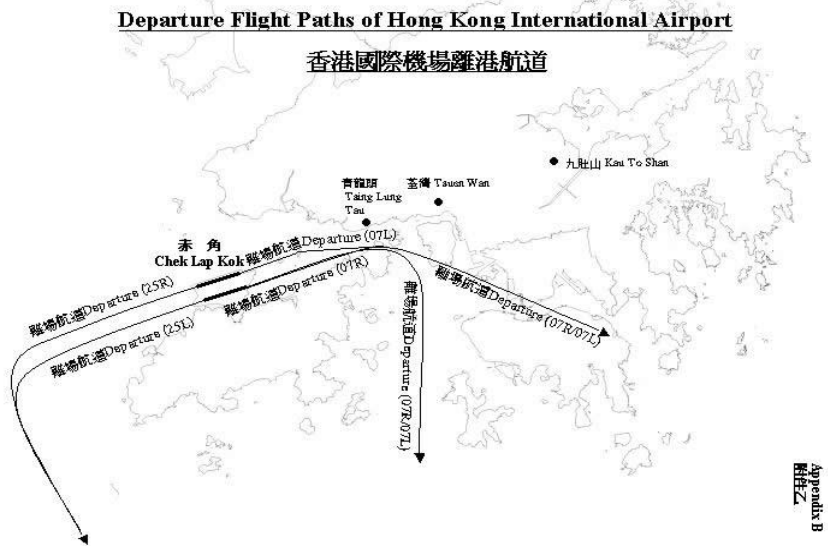
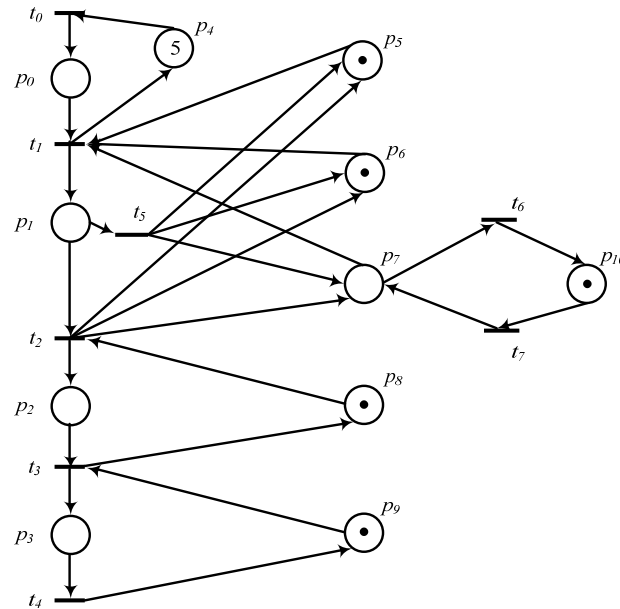


Figure 5.3 Departure flight paths of Hong Kong international airport.

Source: Departure Flight Paths of HK international Airport (2004).
 From http://www.cad.gov.hk/english/ac_path.html accessed October 9, 2010.

5.3 Combined Models

5.3.1 A Departure Model



t_0 : airplane departs permission and enters taxi way

t_1 : airplane enters runway, and starts departure process

t_2 : airplane enters 3-6mile segments

t_3 : airplane enters 6-9mile segments

t_4 : airplane gets departure successfully

t_5 : cancel departure due to emergency

t_6 : inactivate departure flight path

t_7 : activate departure flight path

p_0 : airplane waits for departure

p_1 : airplane enters runway and flies in 0-3mile segments

p_2 : airplane flies in 3-6mile segments

p_3 : airplane flies in 6-9mile segments

p_4 : taxi way buffer

p_5 : -3-0mile departure flight path resource

p_6 : runway resource

p_7 : 0-3mile departure flight path resource

p_8 : 3-6mile departure flight path resource

p_9 : 6-9mile departure flight path resource

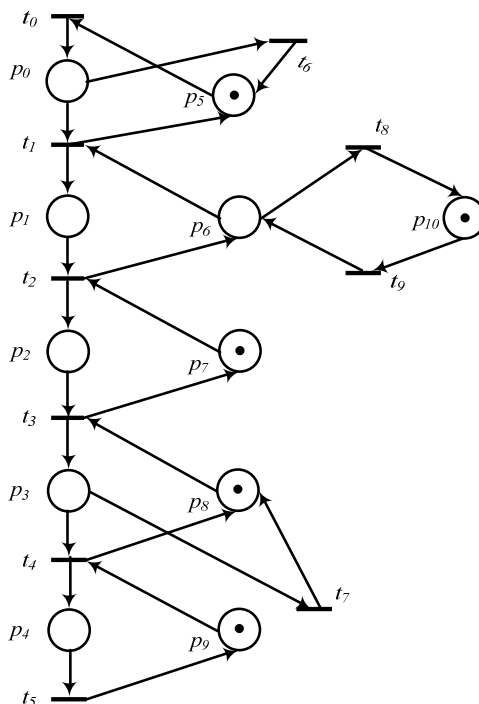
p_{10} : departure flight path resource

Figure 5.4 Petri Net representation of a departure model.

Use a bottom up approach to build an airport tower ATC model. First, build arrival and departure subnets separately, and then combine them to a one-direction subnet including arrival and departure control.

Figure 5.4 shows a departure subnet model. Due to the safety requirement, the distance between any two aircraft is required to be at least three miles in terminal airspace at lower levels. The flight path is divided into several three-mile places. Places p_4 , p_6 , p_7 , and p_8 represent segments of a flight path. p_5 presents the runway. A token in p_4 , p_6 , p_7 , p_8 and p_5 means that the resource is available for an aircraft to enter places p_0 , p_1 , p_2 and p_3 represent the status of an airplane that maintains in a certain segment of its flight path. Firing transitions t_0 , t_1 , t_2 , t_3 , and t_4 and t_5 represent that the airplane enters certain statuses from original statuses. Firing transactions t_6 or t_7 will inactivate or activate departure flight path. The departure model could fully simulate the real departure control in an airport.

5.3.2 An Arrival Model



- | | |
|---|---|
| t0: airplane approaches arrival flight path | p0: airplane approaches arrival flight path |
| t1: airplane enters 6-9mile segments | p1: airplane flies in 6-9mile segments |
| t2: airplane enters 3-6mile segments | p2: airplane flies in 3- 6mile segments |
| t3: airplane enters 0-3mile segments | p3: airplane flies in 0-3mile segments |
| t4: airplane touches down on runway | p4: airplane land on runway |
| t5: airplane lands successfully | p5: airspace for entering arrival flight path |
| t6: airplane takes buffer route | p6: 6-9mile arrival flight path resource |
| t7: airplane bypasses airport | p7: 3-6mile arrival flight path resource |
| t8: inactivate departure flight path | p8: 0-3mile arrival flight path resource |
| t9: activate departure flight path | p9: airport runway resource |
| | p10: arrival flight path resource |

Figure 5.5 Petri net representation of an arrival model.

Figure 5.5 shows the arrival subnet model. It is similar to the departure one. There are two extra transitions (t_6 and t_7) that present the emergency handling process. Transition t_6 fires when an airplane approaches the airport and the arrival flight path is unavailable. The aircraft has to fly away by taking a buffer route. Transition t_7 fires when the aircraft has already entered an arrival flight path and the runway is unavailable for landing due to any emergency, and the aircraft has to cancel its landing operation and bypass the airport.

5.3.3 One-runway one-direction Petri Net Model

By combining the subnets of a departure and arrival model, a One-direction model is constructed as shown in Figure 5.6, which includes both departure and arrival control functions in the same direction. The subnets of departure and arrival are connected by shared resource places.

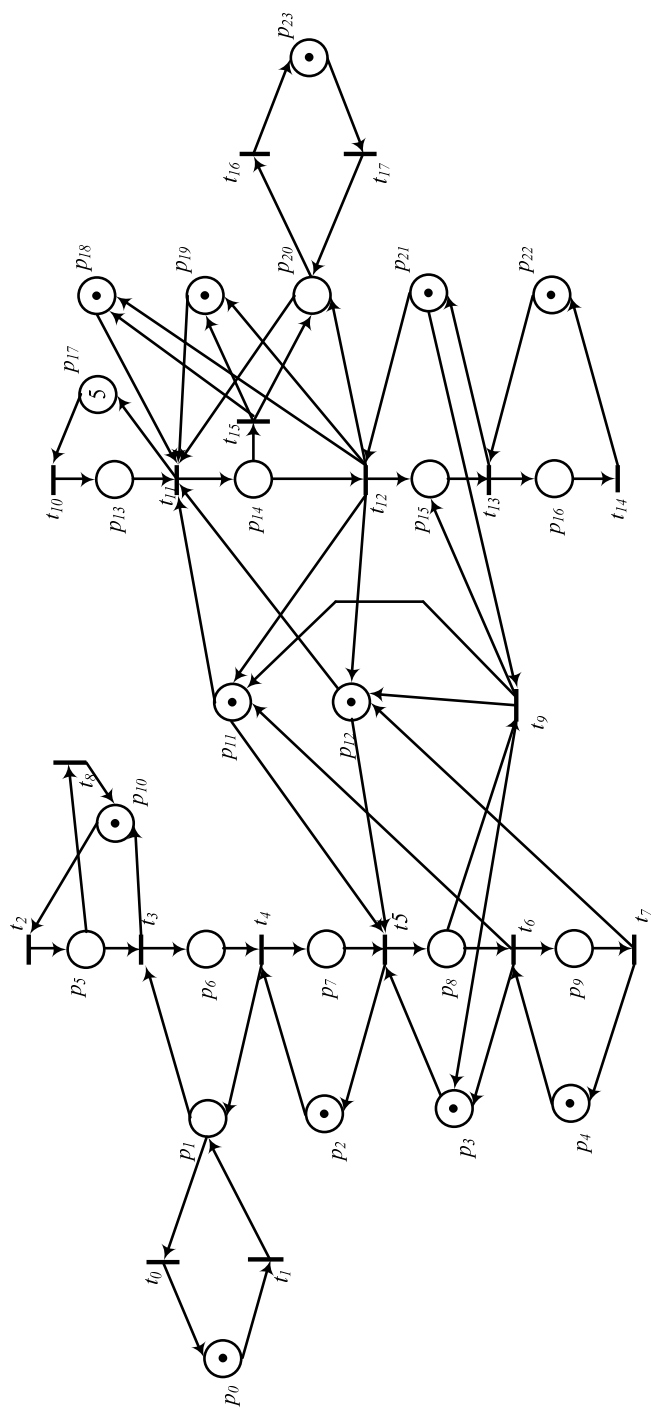


Figure 5.6 One-runway one-direction Petri net model.

As mentioned before, the runway is operated in bi-direction, which means that any direction could be used to land and depart for airplanes. Of course, only one direction can be used at a time. For one runway, the departure and arrival flight paths are along the same direction. By using some extra controls, a Petri net model of bi-direction runway control can be constructed with two one-direction models to make sure only one direction is activated for departure and arrival by aircraft.

The developed model may be used to construct a Petri Net model for any airport's ATC system by using two one-direction models. It can also construct the model of an automatic air traffic control system for a one-runway airport. Using four one-direction Petri nets, it can construct a model of an automatic ATC system for a two-runway airport. Using six one-direction models, it can do so for a three-runway airport.

For example, the Hong Kong International Airport has two runways. In that case, the model of its ATC system can be constructed by using four one-direction Petri. Figure 5.7 shows such a Petri net model.

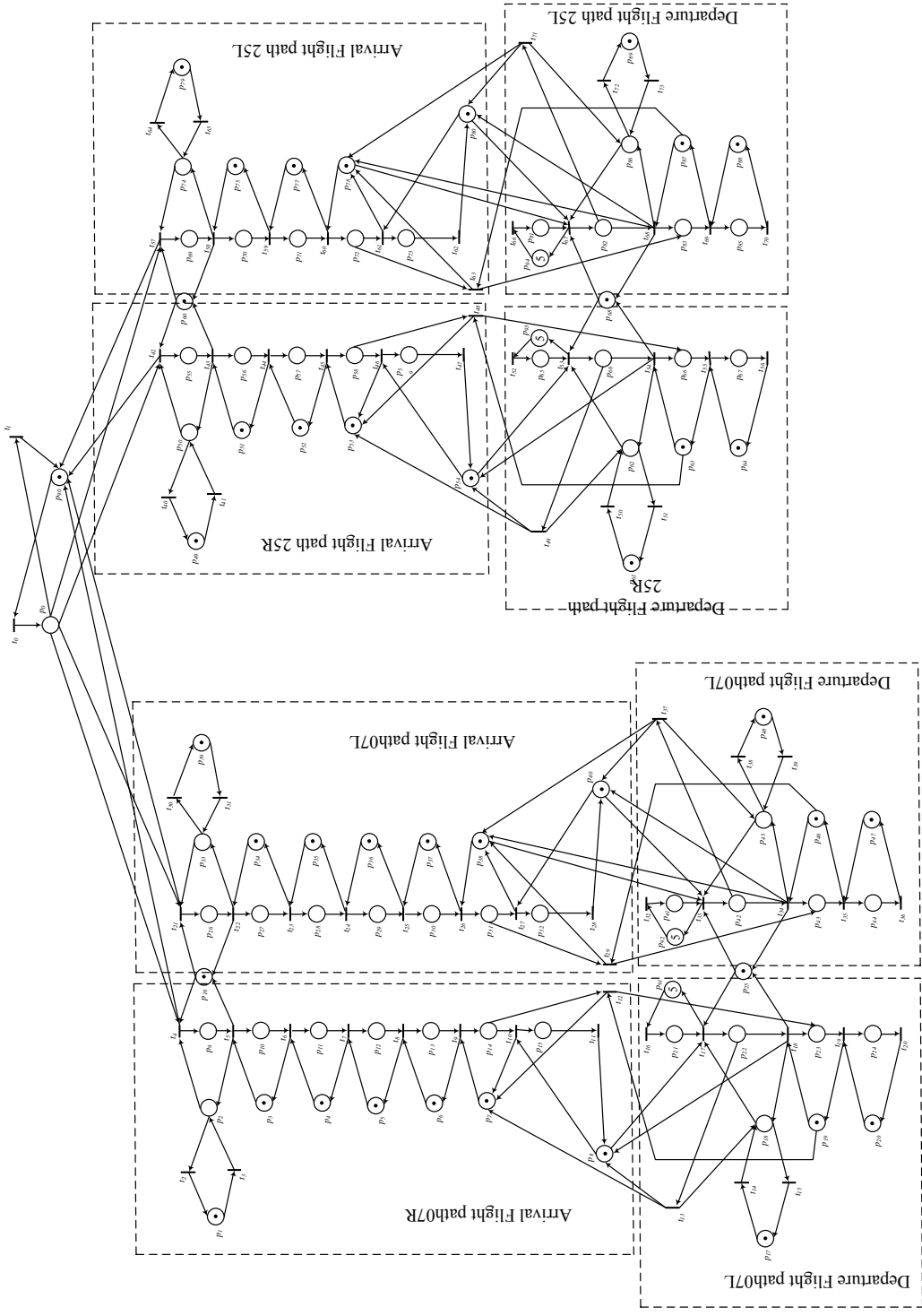


Figure 5.7 Petri Net model for HK International airport ATC system

When a flight finishes its departure process, the flight is going to face an open space in the air and enter the En route stage of its flight process. The flight can choose any direction when there is an air space available. In the real situation, the flight is facing a 3-D space. In order to simplify the problem, a set of assumptions should be made first. It is assumed that an airplane cruise climbs to its desired altitude, and then maintains that altitude until it descends into its destination airport. “Cruise climbing” means that an airplane climbs at an optimal rate to the target altitude without spending time in level flight at intermediate altitudes (Holloway *et al*, 1997). The aircraft can change direction vertically to change altitude. At same time, it will keep the direction in a horizontal plane; or it can change its direction in a horizontal plane and has to keep itself in the same altitude. The 3-D flight can be converted to a two 2-D model (vertical and horizontal).

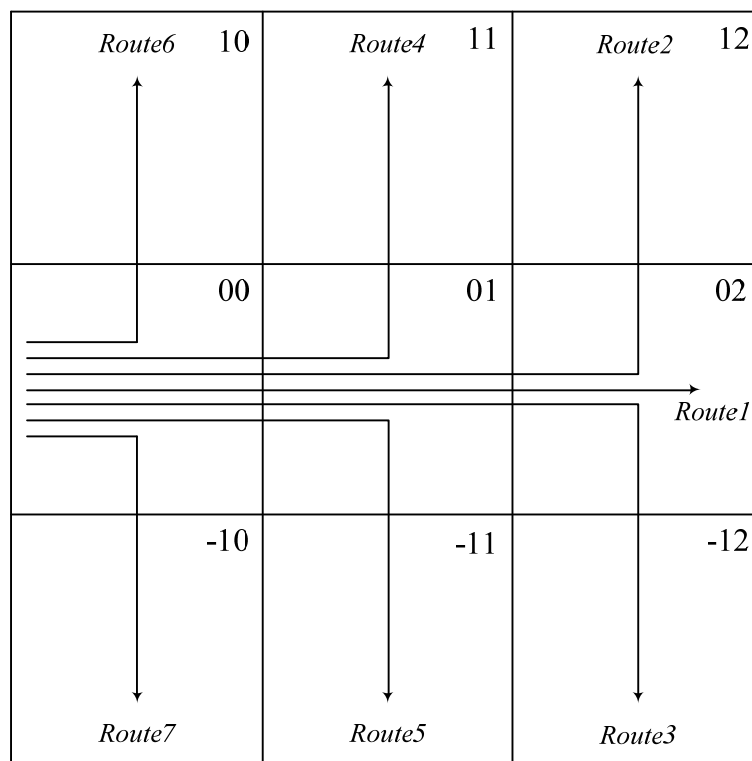


Figure 5.8 2-D view for horizontal operations.

Figure 5.8 gives a 2-D view for horizontal operations. The whole area is 30 miles \times 30 miles, and is divided into nine zones with 10 miles \times 10 miles, which presents the air space that the flight will encounter. Encode them as 00, 01, 02, 10, 11, 12, -10, -11, and -12. In addition, to meet the requirement of air flight separations in an en route stage, it is assumed that the airplane flies through the center of each zone, which ensures that the airplane has at least 5 miles distance away from any other objects in the airspace around.

When an airplane is entering zone 00, the control unit request nine-zone airspace information from the ADS-B database. It determines all available flight routes based on the airspace status. If each of the zones is available, in that case, the airplane has seven available flight routes in total. Those are route S1 - S7. For example, because other objects have already occupied zone 02 is bad, it is not available. In that case, the airplane has four flight routes only, which are route S4 - S7. The control unit sends available flight routes to the GUI module for a pilot to choose.

When the flight is leaving the 30 miles \times 30 miles area, the control unit will be reset and obtain the air space information for the new area from the ADS-B database and determines available flight routes again.

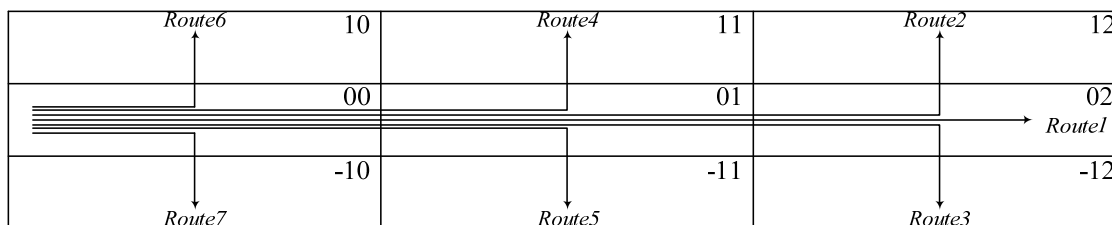


Figure 5.9 2-D view for vertical operations

Figure 5.9 gives a 2-D view for vertical operations. The concept is as same as Figure 5.8's horizontal view. The only difference is the size of each zone. In Figure 5.9, the whole size of the area is 6000 feet \times 30 miles, and each zone is 2000 feet \times 10 miles when the aircraft operate between the surface and an altitude of 29,000 feet; or the whole size of the area is 12000 feet \times 30 miles, and each zone is 4000 feet \times 10 miles when the aircraft operate above 29,000 feet. According to the requirement for vertical separation, between the surface and an altitude of 29,000 feet (8,800 m), no aircraft should come closer vertically than 1,000 feet, and above 29,000 feet (8,800 m) no aircraft shall come closer than 2,000 feet.

The Petri net model is constructed for both vertical and horizontal operations as shown in Figure 5.9.

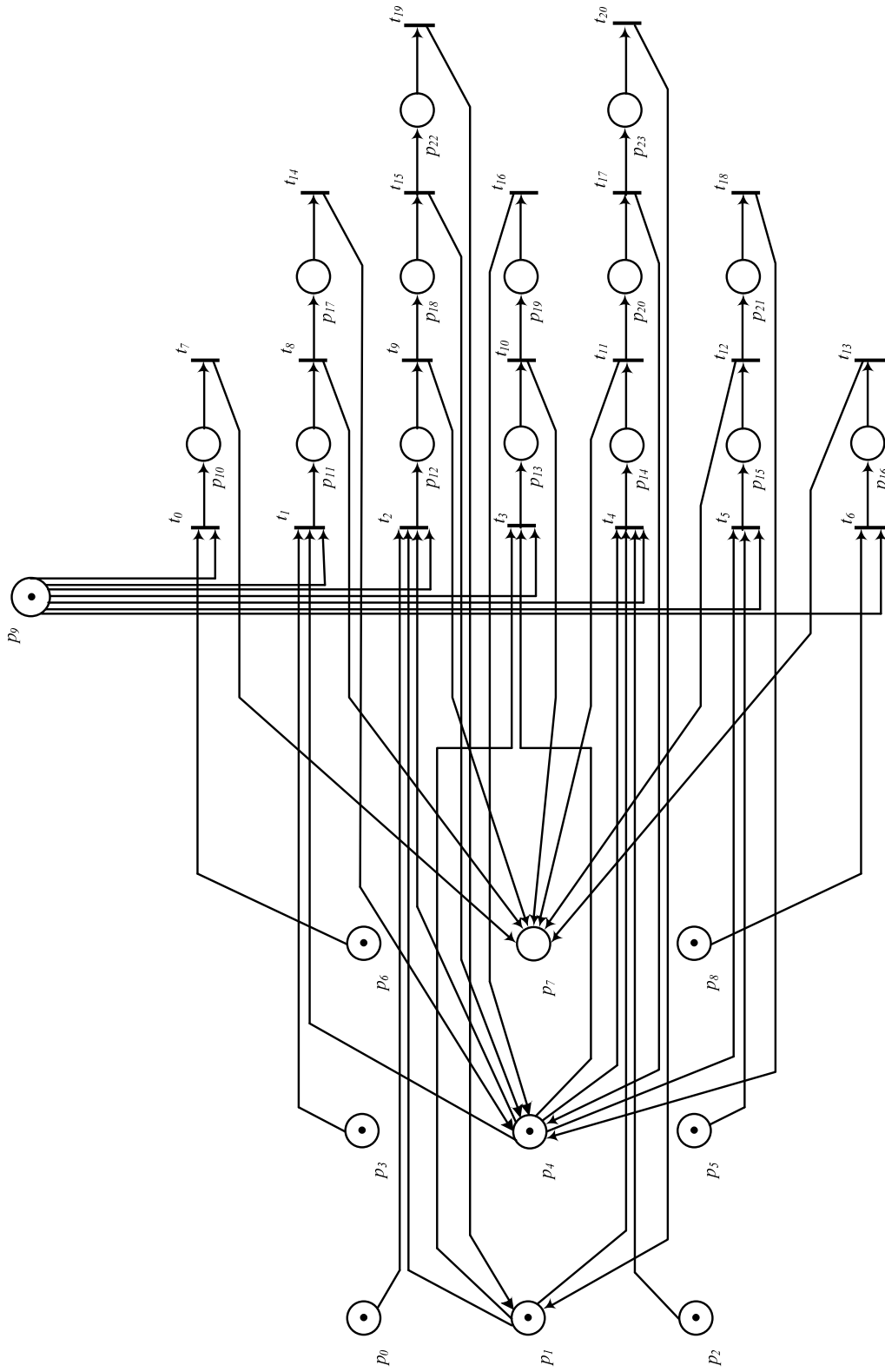


Figure 5.10: Petri Net model for En route control

Table 5.1 Description of en route Petri net Control Model.

t ₀	Airplane chooses route 6		Airplane route request
t ₁	Airplane enters airspace 10 and control unit reset	p ₁	Airspace resource 10 according to 2-D view figure
t ₂	Airplane chooses route 4	p ₂	Airspace resource 11 according to 2-D view figure
t ₃	Airplane enters airspace 01	p ₃	Airspace resource 12 according to 2-D view figure
t ₄	Airplane enters airspace 11 and control unit reset	p ₄	Airspace resource 00 according to 2-D view figure
t ₅	Airplane chooses route 2	p ₅	Airspace resource 01 according to 2-D view figure
t ₆	Airplane enters airspace 01	p ₆	Airspace resource 02 according to 2-D view figure
t ₇	Airplane enters airspace 02	p ₇	Airspace resource -10 according to 2-D view figure
t ₈	Airplane enters airspace 12 and control unit reset	p ₈	Airspace resource -11 according to 2-D view figure
t ₉	Airplane chooses route 1	p ₉	Airspace resource -12 according to 2-D view figure
t ₁₀	Airplane enters airspace 01	p ₁₀	Airplane flies within airspace 00
t ₁₁	Airplane enters airspace 02 and control unit reset	p ₁₁	Airplane flies within airspace 00
t ₁₂	Airplane chooses route 3	p ₁₂	Airplane flies within airspace 01
t ₁₃	Airplane enters airspace 01	p ₁₃	Airplane flies within airspace 00
t ₁₄	Airplane enters airspace 02	p ₁₄	Airplane flies within airspace 01
t ₁₅	Airplane enters airspace -12 and control unit reset	p ₁₅	Airplane flies within airspace 02
t ₁₆	Airplane chooses route 5	p ₁₆	Airplane flies within airspace 00
t ₁₇	Airplane enters airspace 01	p ₁₇	Airplane flies within airspace 01
t ₁₈	Airplane enters airspace -11 and control unit reset	p ₁₈	Airplane flies within airspace 00
t ₁₉	Airplane chooses route 7	p ₁₉	Airplane flies within airspace 01
t ₂₀	Airplane enters airspace -10	p ₂₀	Airplane flies within airspace 02
		p ₂₁	Airplane flies within airspace 00
		p ₂₂	Airplane flies within airspace 01
		p ₂₃	Airplane flies within airspace 00

In Figure 5.10, it is divided into two parts: the air space resource module (left side) and the control module (right side).

The air space resource module communicates with the ADS-B database, reads and updates air space information. Places p_0 , p_1 , p_2 , p_3 , p_4 , p_5 , p_6 , p_7 and p_8 represent air zones in the 2-D view. It means that the air space is available when there is a token in it. Otherwise the air space is unavailable because it is occupied by other airplanes or prohibited by hazard weather and unauthorized air space.

The control module determines all available flight routes based on the air space resource information from the air space resource module. Pilots can review flight routes and make a final decision. After they choose a certain flight route, the control module will pre-occupy certain air space for a selected flight route and update the air space status in ADS-B database at the same time. The air space resource is operated on the first-come, first-serve (FCFS) policy. In that case, if certain air space has been occupied by an airplane, it shows “occupied” or “unavailable” in the control units on any other airplanes that request for the same air space resource. Moreover, the control unit monitors the whole process when that flight route is in use, and updates the air space information when the occupied air space is released as an airplane flies out of it. Transactions t_0 , t_1 , t_2 , t_3 , t_4 , t_5 and t_6 represent all possible flight routes. Firing any one of their respective transitions represents that an airplane enters a certain flight route selected by pilots and consumes tokens from the corresponding air space resources places. Places ($p_{10} - p_{23}$) represent airplane flies in certain air zones corresponding to the 2-D view. An airplane can only enter certain air zones according to the selected flight route, and generate a token back to certain air space resource place when the airplane leaves that air zone. Transactions t_7 , t_{14} , t_{19} , t_{16} , t_{20} , t_{18} and t_{13} represent the control unit that will request new air space information

from ADS-B database and determine a flight route again when aircraft leave the air space area showed in 2-D view.

In real situations, vertical and horizontal operations operate simultaneously to generate control signals for en route airplanes. They use the same en route Petri net control model, but work on different air space resources. When the airplane leaves the ninw-block air zone either in vertical or horizontal operation, both operators will reload air space information accordingly from ADS-B database and determine available flight routes again.

By using the en route Petri net model, the pilot can choose the best flight route and appropriate flight speed with the best fuel efficiency. In addition, the pilot determines the time to enter the arrival flight path of destination airport according to the flight speed, distance to the destination, flight schedule and the arrival operation rate of the destination airport. The pilot requests and reserves a time slot for entering the arrival flight path of the destination airport. All arrival airplanes reserve their unique time slots for entering arrival flight path, which ensure that no potential conflicts occur. These advances will improve the flow of arrival traffic to maximize the use of existing airport facility. In that case, it will make it possible continue using those runways safely, by providing better-defined path assignments and appropriate separation between aircraft (NextGen Descent and Approach, 2010).

CHAPTER 6

SIMULATING AND ANALYSIS

6.1 Simulation via VHDL

In this section, Petri net models are used for simulating of airport control systems. Here we use airport tower control as an example. Petri net analysis software is used to analyze the Petri net model of airport tower control. Through analyzing its behavior, the Petri net module is found to be live, safe and deadlock-free, which means that using the Petri net model to construct the automatic airport traffic control system will achieve the safety requirement. We also use VHDL code to simulate the airport tower control and test the Petri net model.

The airport tower control unit can enable and disable a flight path, showing its resource status to ATC system supervisors and pilots. It can also send out proper regulation command signals: Aapprove-Arrival approval signal for airplane to enter arrival flight path, Dapprove:-Departure approval signal for airplane to enter runway , and bypass to pilots based on the flight path resource status. The control unit includes 3 inputs and 8 outputs. The description of the input and output signals is shown in Table 6.1.

Table 6.1 Description of input and output signals for a control unit

Signal	I/O	Description
Denable	I	Enable departure flight path
Aenable	I	Enable arrival flight path
Radarsignal(8)	I	Airplane waits to departure
Radarsignal(7)	I	Departure flight path 6-9mile segment is occupied
Radarsignal(6)	I	Departure flight path 3-6mile segment is occupied
Radarsignal(5)	I	Departure flight path 0-3mile segment is occupied
Radarsignal(4)	I	Runway is occupied
Radarsignal(3)	I	Arrival flight path 0-3mile segment is occupied
Radarsignal(2)	I	Arrival flight path 3-6mile segment is occupied
Radarsignal(1)	I	Arrival flight path 6-9mile segment is occupied
Radarsignal(0)	I	The airspace for entering arrival flight path is occupied
D6to9resource	O	Departure flight path 6-9mile segment resource status
D3to6resource	O	Departure flight path 3-6mile segment resource status
D0to3resource	O	Departure flight path 0-3mile segment resource status
runwayresource	O	Runway resource status
A0to3resource	O	Arrival flight path 0-3mile segment resource status
A3to6resource	O	Arrival flight path 3-6mile segment resource status
A6to9resource	O	Arrival flight path 6-9mile segment resource status
Dapprove	O	Departure approval signal for airplane to enter runway
Aapprove	O	Arrival approval signal for airplane to enter arrival flight path
Bypass	O	Signal for flight to bypass the airport due to any emergency

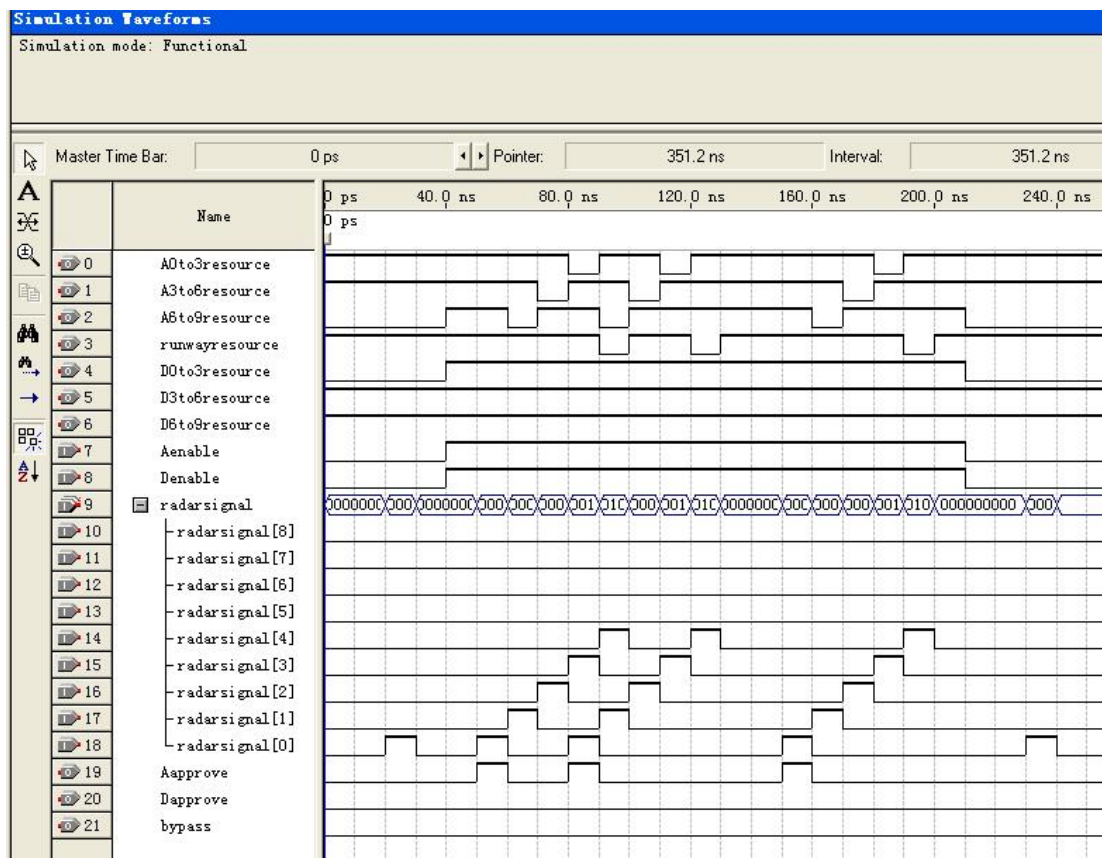


Figure 6.1 Simulation waveform for handling arrival flights.

The simulation reports are discussed as follows. Figure 6.1 shows how the control unit handles arrival flights. Radarsignal (0) indicates that there is a flight approaching the arrival flight path and requesting to land. Without enabling the arrival path, the flight cannot receive approved signal (Approve) to enter the arrival flight path, or it has to fly away. When the arrival flight path is enabled, the flight can receive approved signal and enter the path until it touches down. There are three arrival flights that are successfully landed on this waveform. The signal (0 – 6) in this waveform represents the flight path segments status. Signals 2 and 4 are zero when the arrival and departure flight paths are disabled. A flight path segment is available for an airplane to enter only if its value is ‘1’.

Otherwise, it is disabled or occupied by other airplanes. The flight path resource information can be shared between an ATC supervisor and flight pilot.

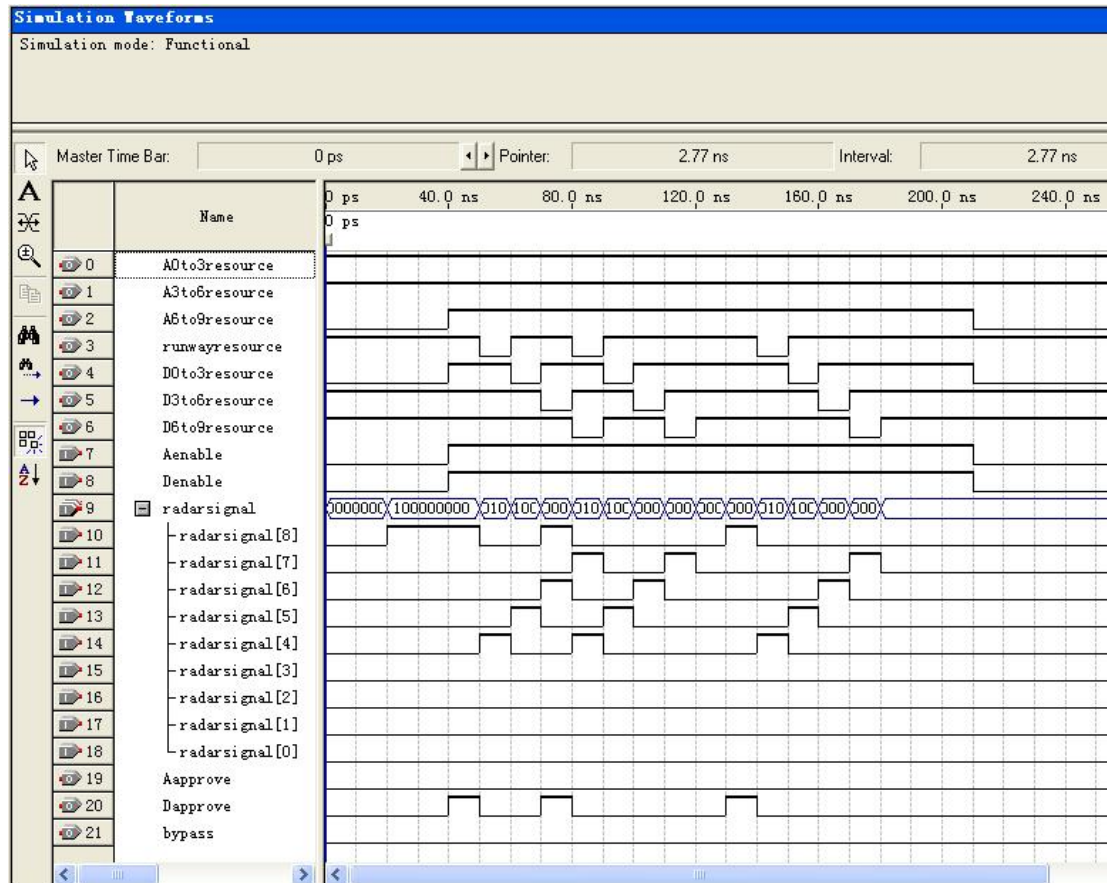


Figure 6.2 Simulation waveform for handling departure flights.

Figure 6.2 shows how the control unit handles the departure flights. It is very similar to Figure 6.1. Radarsignal (8) indicates that there is a flight waiting to depart. Without enabling the departure flights, the flight cannot receive the approved signal to enter runway to take off. It has to wait until the Departure flight path enabled by Denable signal, and then the flight can enter runway to take off and fly away. There are three flights departing successfully based on this waveform.

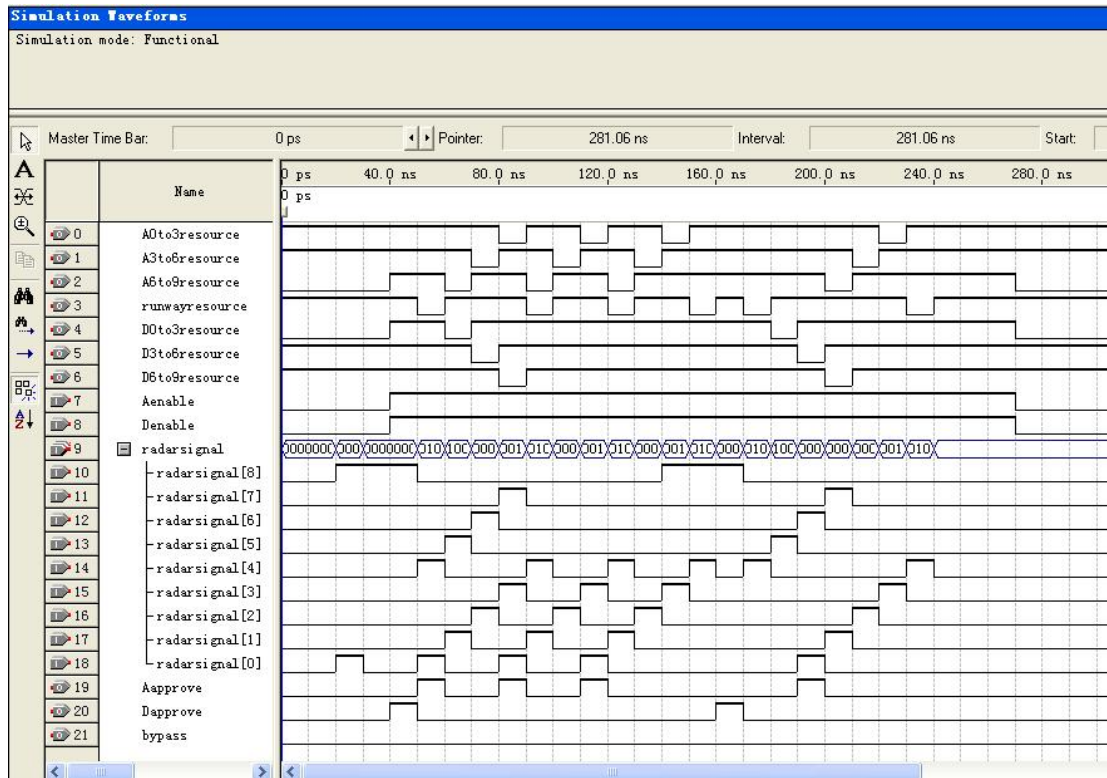


Figure 6.4 Simulation waveform for handling departure and arrival flights simultaneously.

Figure 6.4 shows how the control unit handles the departure and arrival flights at the same time. The control unit handles two departure flights and four arrival flights successfully based on the waveform.

The simulation result shows that the Petri nets model can work properly to handle the arrival and departure aircraft for the normal airport operation and some emergency cases.

6.2 Analysis via INA

In this section, INA is used to analyze of the properties of Petri net models including the boundedness, liveness and safeness. INA is a software tool supporting the analysis of Petri nets. In terms of Petri nets, liveness means that a Petri net can always evolve starting from any state to allow a modeled event/activity to be performed. The physical meaning of this property is that no deadend can occur for the entire physical system. Such a property is a prerequisite; otherwise, the system may come to a state under which no airplane can move. Safeness means the token in each place cannot be greater than one. Physically, this means that no two airplanes can run on the same way or occupy the same segment of space at the same time. Obviously, this property is crucial; otherwise, collision may occur. Boundness means that the number of tokens in each place is limited by a finite number. This property physically ensures that overflow is avoided at any stage.

Arrival Petri net model:

The INA analysis results show that the Arrival Petri net model is ordinary, homogenous and bounded. In addition, it has no dead transitions and no dead reachable states. The net is live and safe as shown in Fig. 6.5.

```

C:\Users\hang.wu\Desktop\INA\INAwin32.exe
Analysis menu:
Decide structural boundedness.....B
Non-reachability test of a partial marking using the state equation.....N
Compute the symmetries of the net.....Y

Compute a shortest path from the initial state to a target marking.....P
Compute a minimal path from the initial state to satisfy a predicate.....O
Compute a reachability graph.....R
Compute a coverability graph to decide boundedness and coverability.....G

Compute a basis for all P/T-invariants [non-reachability test].....I
Compute a basis for all semipositive P/T-[sub/sur]-invariants.....S
Format lines written to INUARI.HLP earlier.....F
Test place- or transition-vectors for invariant properties.....T

Check the deadlock-trap-, SMD-, SMA-properties [boundedness, liveness].....D
Compute all components [check SMD-,SMA-properties].....C
Decide state machine coverability [for boundedness].....M

choice > B
Deciding structural boundedness
The net is covered by semipositive place sub-invariants.
Computation of the reachability graph
States generated:      48
The net has no dead transitions at the initial marking.
The net has no dead reachable states.
The net is safe.
Liveness test:
Computing the strongly connected components
The net is live.
The net is live, if dead transitions are ignored.
The net is live and safe.
The net is covered by semipositive T-invariants.
We check the deadlock-trap-property.

processed candidates:      31
The deadlock-trap-property is valid.
The net is state machine decomposable (SMD).
The net is coverable by state machines (SMC).
The net is covered by semipositive P-invariants.
Executing SMA-test.....
The net is state machine allocatable (SMA).
The net is covered by 1-token SCSM's.
ORD HOM NBM PUR CSU SCF CON SC Ft0 tF0 Fp0 pF0 MG SM FC EFC ES
Y Y Y Y N Y Y N N N N N N N N
DTP SMC SMD SMA CPI CTI B SB REU DSt BSt DTr DCF L LU L&S
Y Y Y Y Y Y Y ? N ? N ? Y Y Y

```

Figure 6.5 INA analysis result for an arrival model.

Departure Petri net model:

The INA analysis shows that the departure Petri net model is bounded and live as shown in Fig. 6.6. The net is not safe because the total number of tokens in place p4 is five. The taxi way buffer can support up to five airplanes that are waiting for taking off.

```

C:\Users\hang.wu\Desktop\INA\INAwin32.exe
Analysis menu:
Decide structural boundedness.....B
Non-reachability test of a partial marking using the state equation.....N
Compute the symmetries of the net.....Y

Compute a shortest path from the initial state to a target marking.....P
Compute a minimal path from the initial state to satisfy a predicate.....O
Compute a reachability graph.....R
Compute a coverability graph to decide boundedness and coverability.....G

Compute a basis for all P/T-invariants [non-reachability test].....I
Compute a basis for all semipositive P/T-[sub/sur]-invariants.....S
Format lines written to INVARI.HLP earlier.....F
Test place- or transition-vectors for invariant properties.....T

Check the deadlock-trap-, SMD-, SMA-properties [boundedness, liveness].....D
Compute all components [check SMD-,SMA-properties].....C
Decide state machine coverability [for boundedness].....M

choice > B
Deciding structural boundedness
eliminated columns:      1
The net is structurally bounded.
The net is bounded.
There are no proper semipositive I-surinvariants.
The net is covered by semipositive place sub-invariants.
Computation of the reachability graph
States generated:      72
The net has no dead transitions at the initial marking.
The net has no dead reachable states.
choice> L
Liveness test:
Computing the strongly connected components
The net is live.
The net is live, if dead transitions are ignored.
The net is covered by semipositive I-invariants.
We check the deadlock-trap-property.

processed candidates:      35
The deadlock-trap-property is valid.
The net is state machine decomposable (SMD).
The net is coverable by state machines (SMC).
The net is covered by semipositive P-invariants.
Executing SMA-test.....
The net is state machine allocatable (SMA).
The net is not covered by 1-token SCSM's.
ORD HOM NBM PUR CSU SCF COM SC Ft0 tF0 Fp0 pF0 MG SM FC EFC ES
Y Y Y Y N N Y Y N N N N N N N Y
DIP SMC SMD SMA CPI CII B SB REU DSt BSt DTr DCF L LU L&S
Y Y Y Y Y Y Y Y ? N ? N ? Y Y N

```

Figure 6.6 INA analysis result for a departure model.

One-direction Petri net model:

The INA analysis shows that the one-direction Petri net model is bounded and live as shown in Fig. 6.7. One-direction Petri net model is integrated models. Since it is reachability-base analysis, one may face computational challenges due to state explosion problems. Hence, more efficient methods, e.g., reduction (Li, 2008), need to be pursued.



```

E:\INA\INAwin32.exe
Analysis menu:
Decide structural boundedness.....B
Non-reachability test of a partial marking using the state equation.....N
Compute the symmetries of the net.....Y

Compute a shortest path from the initial state to a target marking.....P
Compute a minimal path from the initial state to satisfy a predicate.....O
Compute a reachability graph.....R
Compute a coverability graph to decide boundedness and coverability.....G

Compute a basis for all P/T-invariants [non-reachability test].....I
Compute a basis for all semipositive P/T-[sub/sur]-invariants.....S
Format lines written to INVARI.HLP earlier.....F
Test place- or transition-vectors for invariant properties.....I

Check the deadlock-trap-, SMD-, SMA-properties [boundedness, liveness].....D
Compute all components [check SMD-, SMA-properties].....C
Decide state machine coverability [for boundedness].....M

choice > B
Deciding structural boundedness
eliminated columns:      1
The net is structurally bounded.
The net is bounded.
There are no proper semipositive T-surinvariants.
The net is covered by semipositive place sub-invariants.
choice > R
Current analysis options are:
  no symmetrical reduction
  no stubborn reduction
  no depth restriction
  do not use a 'bad' predicate
.....Reset options? Y/N N
Computation of the reachability graph
States generated:      2592
The net has no dead transitions at the initial marking.
The net has no dead reachable states.
Liveness test:
Computing the strongly connected components
The net is live.
The net is live, if dead transitions are ignored.
The net is covered by semipositive I-invariants.
We check the deadlock-trap-property.

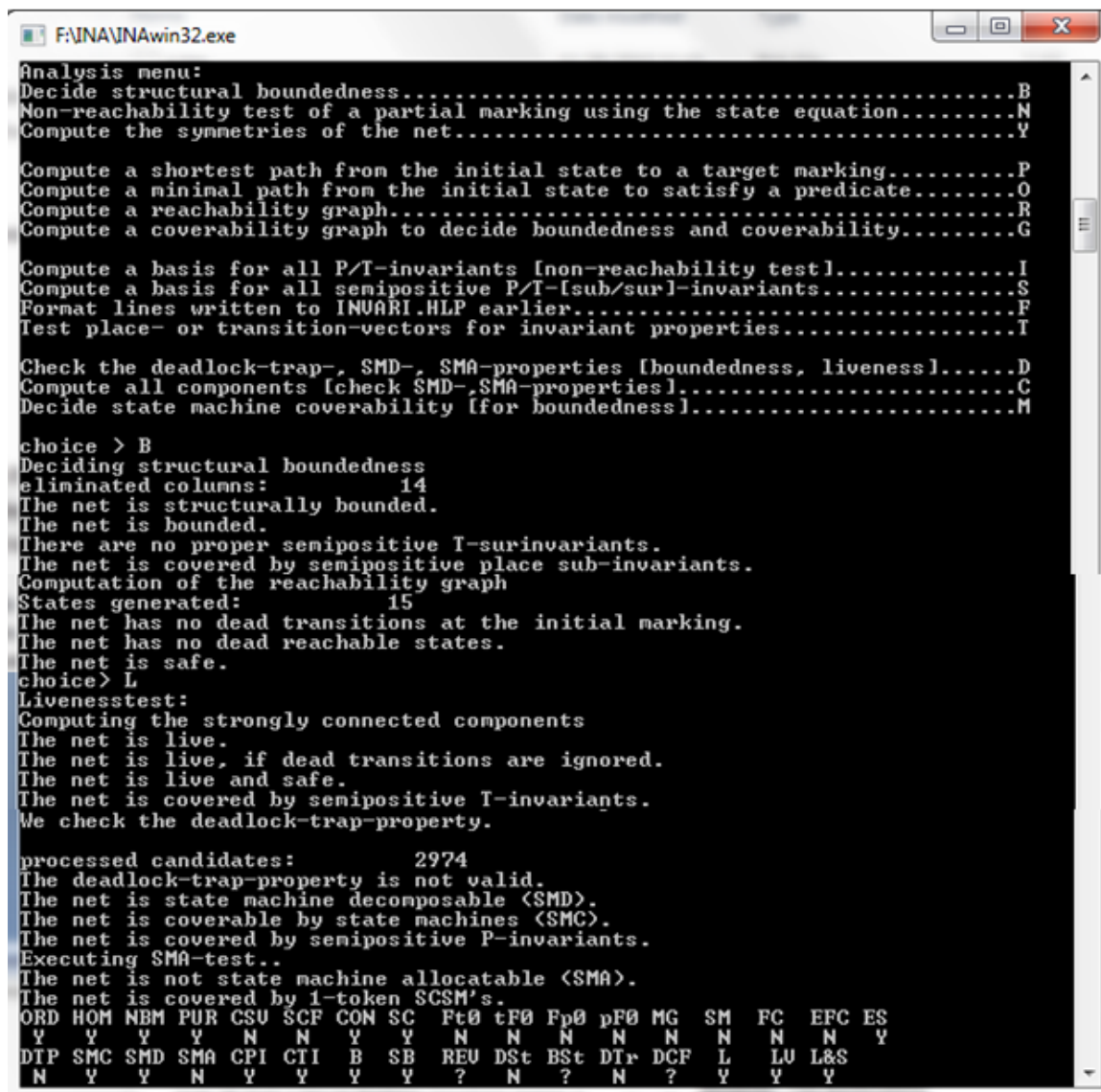
processed candidates:      228
The deadlock-trap-property is valid.
The net is state machine decomposable (SMD).
The net is coverable by state machines (SMC).
The net is covered by semipositive P-invariants.
Executing SMA-test
The net is not state machine allocatable (SMA).
The net is not covered by 1-token SCSM's.
ORD HOM NBM PUR CSU SCF CON SC Ft0 tF0 Fp0 pF0 MG SM FC EFC ES
Y Y Y Y N N Y Y N N N N N N N
DTP SMC SMD SMA CPI CTI B SB REU DSt BSt DTr DCF L LU L&S
Y Y Y N Y Y Y Y ? N ? N ? Y Y N

```

Figure 6.7 INA analysis result for one-direction model.

En route Petri net model:

The INA analysis shows that en route Petri net model is bounded, live and safe as shown in Fig. 6.8. It has no dead transitions at the initial marking and has no dead reachable states.



```

F:\INA\INAwin32.exe
Analysis menu:
Decide structural boundedness.....B
Non-reachability test of a partial marking using the state equation.....N
Compute the symmetries of the net.....Y

Compute a shortest path from the initial state to a target marking.....P
Compute a minimal path from the initial state to satisfy a predicate.....O
Compute a reachability graph.....R
Compute a coverability graph to decide boundedness and coverability.....G

Compute a basis for all P/T-invariants [non-reachability test].....I
Compute a basis for all semipositive P/T-[sub/sur]-invariants.....S
Format lines written to INVARI.HLP earlier.....F
Test place- or transition-vectors for invariant properties.....I

Check the deadlock-trap-, SMD-, SMA-properties [boundedness, liveness].....D
Compute all components [check SMD-,SMA-properties].....C
Decide state machine coverability [for boundedness].....M

choice > B
Deciding structural boundedness
eliminated columns:      14
The net is structurally bounded.
The net is bounded.
There are no proper semipositive I-surinvariants.
The net is covered by semipositive place sub-invariants.
Computation of the reachability graph
States generated:      15
The net has no dead transitions at the initial marking.
The net has no dead reachable states.
The net is safe.
choice> L
Livenessstest:
Computing the strongly connected components
The net is live.
The net is live, if dead transitions are ignored.
The net is live and safe.
The net is covered by semipositive I-invariants.
We check the deadlock-trap-property.

processed candidates:      2974
The deadlock-trap-property is not valid.
The net is state machine decomposable (SMD).
The net is coverable by state machines (SMC).
The net is covered by semipositive P-invariants.
Executing SMA-test..
The net is not state machine allocatable (SMA).
The net is covered by 1-token SCSM's.
ORD HOM NBM PUR CSU SCF CON SC Ft0 tF0 Fp0 pF0 MG SM FC EFC ES
Y Y Y Y N N Y Y N N N N N N N N Y
DTP SMC SMD SMA CPI CTI B SB REU DSt BSt DTp DCF L LU L&S
N Y Y N Y Y Y Y ? N ? N ? Y Y Y

```

Figure 6.8 INA analysis result for en route model.

CHAPTER 7

CONCLUSION

Dealing with air traffic control is so time consuming and complex that large planes require multiple crewmen, and single-pilot planes have many restrictions and where and when they can fly. The new air traffic control system, the distribution of controller workload will shift away from monitoring the separations of all aircraft in a sector and toward the management of traffic flow and handling those exceptional problems that only humans have the knowledge and skill to solve (Rocco, 2001).

In this Thesis, it explains the current air traffic control system and the problem it is facing. Petri net models are used to construct the discrete event models at airport tower control systems and En route control systems. The airport tower control system can provide automatic air flow control for arrival and departure flights. The En route control system works on any airplane that operates in En route stage to generate air flow control signals.

In addition, The Petri nets models that are constructed for airport tower control and en route control ensure the safeness, liveness, and deadlock freeness of the air traffic control system. They can provide the safe air traffic without causing any collision. Petri net control models could be used to construct the next generation ATC system to meet the requirement for Free Flight which gives pilots increased flexibility to choose and modify their routes in real time, in order to reduce costs and increase system capacity. Moreover, the Petri net models that are constructed can to some extent replace ATC

controllers and also decrease their workload. In that case, the new Petri net models can guarantee more efficient flow of traffic.

However, there is some work need to be done in the future, including define certain algorithm on En route control to find the best flight route automatically, which makes the automatic air traffic control true.

APPENDIX A

VHDL CODE FOR AIRPORT TOWER ATC CONTROL UNIT

The VHDL code for airport tower ATC control unit is provided as follows.

```
library ieee;
use ieee.std_logic_1164.all;

entity ATC2 is
port ( Denable, Aenable :in bit;
      radarsignal : in bit_vector ( 8 downto 0);

      D6to9resource: out bit;
      D3to6resource: out bit;
      D0to3resource: out bit;
      runwayresource: out bit;
      A0to3resource: out bit;
      A3to6resource: out bit;
      A6to9resource: out bit;

      Dapprove: out bit;
      Aapprove: out bit;
      bypass: out bit);
end ATC2;

architecture behaviral of ATC2 is

    shared variable D6to9 : bit := '1';
    shared variable D3to6 : bit := '1';
    shared variable D0to3 : bit := '0';
    shared variable runway : bit := '1';
    shared variable A0to3 : bit := '1';
    shared variable A3to6 : bit := '1';
```

```

        shared variable DA: bit := '0';
        shared variable AA : bit := '0';
        shared variable bp : bit :='0';

begin
D6to9resource <= D6to9;
D3to6resource <= D3to6;
D0to3resource <= D0to3;
runwayresource <= runway;
A0to3resource <= A0to3;
A3to6resource <= A3to6;
A6to9resource <= A6to9;

Dapprove <= DA;
Aapprove <= AA;
bypass <= bp;
p1 : process (Denable)
begin
if Denable = '1' then
D0to3 := '1';
else
D0to3 := '0';
end if;
if Aenable = '1' then
A6to9 := '1';
else
A6to9 := '0';
end if;
if radarsignal(8) = '1' and runway = '1' and D0to3 = '1' and A0to3 = '1' then
DA := '1';
else
DA := '0';
end if;

```

```
if radarsignal(0) = '1' and A6to9 = '1' then
```

```
AA := '1';
```

```
else
```

```
AA := '0';
```

```
end if;
```

```
if radarsignal(3) = '1' and runway = '0' then
```

```
bp := '1';
```

```
else
```

```
bp := '0';
```

```
end if;
```

```
if radarsignal(1) = '0' and Aenable = '1' then
```

```
A6to9 := '1';
```

```
else
```

```
A6to9 := '0';
```

```
end if;
```

```
if radarsignal(2) = '1' then
```

```
A3to6 := '0';
```

```
else
```

```
A3to6 := '1';
```

```
end if;
```

```
if radarsignal(3) = '1' then
```

```
A0to3 := '0';
```

```
else
```

```
A0to3 := '1';
```

```
end if;
```



```
if radarsignal(4) = '1' then
runway := '0';
else
runway := '1';
end if;
```

```
if radarsignal(7) = '1' then
D6to9 := '0';
else
D6to9 := '1';
end if;
```

```
if radarsignal(6) = '1' then
D3to6 := '0';
else
D3to6 := '1';
end if;
```

```
if radarsignal(5)='0' and Denable = '1' then
D0to3 := '1';
else
D0to3 := '0';
end if;
```

```
end process;
end behavioral;
```

APPENDIX B

INA SOURCE CODE FOR AN ARRIVAL PETRI NET MODEL

The following shows INA source code for an arrival Petri net model.

```

PLACE      M      PRESET  POSTSET      NETZ: arrival
0          0        0      1 6
1          0        1      2
2          0        2      3
3          0        3      4 7
4          0        4      5
5          1        1 6    0
6          0        2 9    1 8
7          1        3      2
8          1        4 7    3
9          1        5      4
10         1        8      9

```

```

@
-----
place nr.  name                                     capacity  time
0: airplane approaches arrival flight path         oo        0
1: airplane flies in 6-9mile segments              oo        0
2: airplane flies in 3- 6mile segments             oo        0
3: airplane flies in 0-3mile segments              oo        0
4: airplane land on runway                        oo        0
5: airspace for entering arrival flight path       oo        0
6: 6-9mile arrival flight path resource            oo        0
7: 3-6mile arrival flight path resource           oo        0
8: 0-3mile arrival flight path resource            oo        0
9: airport runway resource                         oo        0
10: arrival flight path resource                   oo        0

```

```

@
trans nr.  name                                     priority  time
0: airplane approaches arrival flight path         0         0
1: airplane enters 6-9mile segments                0         0
2: airplane enters 3-6mile segments                0         0
3: airplane enters 0-3mile segments                0         0
4: airplane touches down on runway                 0         0
5: airplane lands successfully                      0         0
6: airplane takes buffer route                     0         0
7: airplane bypasses airport                       0         0
8: inactivate departure flight path                 0         0
9: activate departure flight path                    0         0

```

@

APPENDIX C

INA SOURCE CODE FOR A DEPARTURE PETRI NET MODEL

The following shows INA source code for a departure Petri net model.

```

PLACE ..... M ..... PRESET ..... POSTSET ..... NETZ: arrival .....
0 ..... 0 ..... 0 ..... 1 6 .....
1 ..... 0 ..... 1 ..... 2 .....
2 ..... 0 ..... 2 ..... 3 .....
3 ..... 0 ..... 3 ..... 4 7 .....
4 ..... 0 ..... 4 ..... 5 .....
5 ..... 1 ..... 1 6 ..... 0 .....
6 ..... 0 ..... 2 9 ..... 1 8 .....
7 ..... 1 ..... 3 ..... 2 .....
8 ..... 1 ..... 4 7 ..... 3 .....
9 ..... 1 ..... 5 ..... 4 .....
10 ..... 1 ..... 8 ..... 9 .....

@
place nr. name capacity time
0: airplane approaches arrival flight path oo 0
1: airplane flies in 6-9mile segments oo 0
2: airplane flies in 3- 6mile segments oo 0
3: airplane flies in 0-3mile segments oo 0
4: airplane land on runway oo 0
5: airspace for entering arrival flight path oo 0
6: 6-9mile arrival flight path resource oo 0
7: 3-6mile arrival flight path resource oo 0
8: 0-3mile arrival flight path resource oo 0
9: airport runway resource oo 0
10: arrival flight path resource oo 0

@
trans nr. name priority time
0: airplane approaches arrival flight path 0 0
1: airplane enters 6-9mile segments 0 0
2: airplane enters 3-6mile segments 0 0
3: airplane enters 0-3mile segments 0 0
4: airplane touches down on runway 0 0
5: airplane lands successfully 0 0
6: airplane takes buffer route 0 0
7: airplane bypasses airport 0 0
8: inactivate departure flight path 0 0
9: activate departure flight path 0 0

@

```

PLACE	M	PRESET	POSTSET	NETZ: departure
0	0	0	.	1
1	0	1	.	5 2
2	0	2	.	3
3	0	3	.	4
4	5	1	.	0
5	1	5 2	.	1
6	1	5 2	.	1
7	0	5 2 7	.	1 6
8	1	3	.	2
9	1	4	.	3
10	1	6	.	7

@

place nr.	name	capacity	time
0:	airplane waits for departure	oo	0
1:	airplane enters runway and flies in 0-3mile segments	oo	0
2:	airplane flies in 3-6mile segments	oo	0
3:	airplane flies in 6-9mile segments	oo	0
4:	taxi way buffer	oo	0
5:	-3-0mile departure flight path resource	oo	0
6:	runway resource	oo	0
7:	0-3mile departure flight path resource	oo	0
8:	3-6mile departure flight path resource	oo	0
9:	6-9mile departure flight path resource	oo	0
10:	departure flight path resource	oo	0

@

trans nr.	name	priority	time
0:	airplane departs permission and enters taxi way	0	0
1:	airplane enters runway, and starts departure process	0	0
2:	airplane enters 3-6mile segments	0	0
3:	airplane enters 6-9mile segments	0	0
4:	airplane gets departure successfully	0	0
5:	cancel departure due to emergency	0	0
6:	inactivate departure flight path	0	0
7:	activate departure flight path	0	0

@

APPENDIX D

INA SOURCE CODE FOR A ONE-DIRECTION PETRI NET MODEL

The following show INA source code for a one-direction Petri net model.

```

PLACE          M      PRESET,   POSTSET  NETZ:one-direction
0              1        0,      1
1              0        1  4,    0  3
2              1        5,      4
3              1        6  9,    5
4              1        7,      6
5              0        2,      3  8
6              0        3,      4
7              0        4,      5
8              0        5,      6  9
9              0        6,      7
10             1        3  8,    2
11             1        6  9 12,  5  11
12             1        7  9 12,  5  11
13             0        10,     11
14             0        11,     12  15
15             0        9  12,   13
16             0        13,     14
17             5        11,     10
18             1        15 12,   11
19             1        15 12,   11
20             0        12 15 17, 11  16
21             1        13,     12  9
22             1        14,     13
23             1        16,     17
@

```

place nr.	name	capacity	time
0:	arrival flight path resource	oo	0
1:	6-9mile arrival flight path resource	oo	0
2:	3-6mile arrival flight path resource	oo	0
3:	0-3mile arrival flight path resource	oo	0
4:	airport runway resource	oo	0
5:	airplane approaches arrival flight path	oo	0
6:	airplane flies in 6-9mile segments	oo	0
7:	airplane flies in 3-6mile segments	oo	0
8:	airplane flies in 0-3mile segments	oo	0
9:	airplane land on runway	oo	0
10:	airspace for entering arrival flight path	oo	0
11:	0-3mile arrival flight path resource	oo	0
12:	airport runway resource	oo	0
13:	airplane waits for departure	oo	0
14:	airplane enters runway and flies in 0-3mile segments	oo	0
15:	airplane flies in 3-6mile segments	oo	0
16:	airplane flies in 6-9mile segments	oo	0
17:	taxi way buffer	oo	0

APPENDIX E

INA SOURCE CODE MODEL FOR EN ROUTE PETRI NET MODEL

The following show INA source code for en route Petri net model.

```

PLACE      M          1  4   8  11  15  18  20 ,      0      POSTSET      9  12  NETZ: en route
0          1          1  4   8  11  15  18  20 ,      0      2  5  9  12  16  19
1          1          1  4   8  11  15  18  20 ,      0
2          1          1  4   8  11  15  18  20 ,      0
3          1          1  4   8  11  15  18  20 ,      0
4          0          3  6  10  13  17 ,      4  8  11  15  18
5          1          4  7  11  14  18 ,      2  5  9  12  16
6          1          8  15  11 ,      5  9  12
7          1          20 ,      19
8          1          18 ,      16
9          1          15 ,      12
10         0          0 ,      1
11         0          2 ,      3
12         0          3 ,      4
13         0          5 ,      6
14         0          6 ,      7
15         0          7 ,      8
16         0          9 ,      10
17         0          10 ,      11
18         0          12 ,      13
19         0          13 ,      14
20         0          14 ,      15
21         0          16 ,      17
22         0          17 ,      18
23         0          19 ,      20
@

```

place nr.	name	capacity	time
0:	Airplane route request	oo	0
1:	Airspace resource 10	oo	0
2:	Airspace resource 11	oo	0
3:	Airspace resource 12	oo	0
4:	Airspace resource 00	oo	0
5:	Airspace resource 01	oo	0
6:	Airspace resource 02	oo	0
7:	Airspace resource -10	oo	0
8:	Airspace resource -11	oo	0
9:	Airspace resource -12	oo	0
10:	Airplane flies within airspace 00	oo	0
11:	Airplane flies within airspace 00	oo	0
12:	Airplane flies within airspace 01	oo	0
13:	Airplane flies within airspace 00	oo	0
14:	Airplane flies within airspace 01	oo	0
15:	Airplane flies within airspace 02	oo	0
16:	Airplane flies within airspace 00	oo	0
17:	Airplane flies within airspace 01	oo	0

```

18: Airplane flies within airspace 00          oo  0
19: Airplane flies within airspace 01          oo  0
20: Airplane flies within airspace 02          oo  0
21: Airplane flies within airspace 00          oo  0
22: Airplane flies within airspace 01          oo  0
23: Airplane flies within airspace 00          oo  0
@
trans nr.      name                                     priority time
0: Airplane chooses route 6                          0  0
1: Airplane enters airspace 10 and request control unit reset 0  0
2: Airplane chooses route 4                          0  0
3: Airplane enters airspace 01                      0  0
4: Airplane enters airspace 11 and request control unit reset 0  0
5: Airplane chooses route 2                          0  0
6: Airplane enters airspace 01                      0  0
7: Airplane enters airspace 02                      0  0
8: Airplane enters airspace 12 and request control unit reset 0  0
9: Airplane chooses route 1                          0  0
10: Airplane enters airspace 01                     0  0
11: Airplane enters airspace 02 and request control unit reset 0  0
12: Airplane chooses route 3                        0  0
13: Airplane enters airspace 01                    0  0
14: Airplane enters airspace 02                    0  0
15: Airplane enters airspace -12 and request control unit reset 0  0
16: Airplane chooses route 5                       0  0
17: Airplane enters airspace 01                    0  0
18: Airplane enters airspace -11 and request control unit reset 0  0
19: Airplane chooses route 7                       0  0
20: Airplane enters airspace -10                   0  0
@

```

REFERENCES

- Azuma, Ronald. (1999). Visualization of Conflicts and Resolutions in a “Free Flight ” Scenario.
- David, R. & Alla, H. (1994). Petri Nets for modeling of dynamic systems. *Automatica*, 30(2): 175-202.
- Diana, A. & Giua, C. Seatzu. (2001). Safeness-Enforcing Supervisory Control for Railway Networks: Advanced Intelligent Mechatronics, 99-104.
- Hruz, B. & Zhou, M.C. (2007). Modeling and Control of Discrete-event Dynamic Systems: with Petri Nets and Other Tools.
- Holloway, L. E. & Krogh, B. H. & Giua, A. (1997). A survey of Petri net methods for controlled discrete event systems: *Discrete Event Systems*, vol. 7, (pp. 151–190).
- Li, J. (2008). Refining and Verifying Regular Petri Nets. *International Journal of Systems Science*, 39(1), 17-27.
- NASA Air Traffic Management (2000). Retrieved October 20, 2010, from <http://virtualskies.arc.nasa.gov/atm/2.html>.
- Nordwall, Bruce D. (1995). Free Flight: ATC Model for the Next 50 years. *Aviation Week and Space Technology*
- Perry, Tekla S. (1997). In Search of the Future of Air Traffic Control. *IEEE Spectrum*, 18-35
- Rocco, Pamela (2001). Air Traffic Control Specialist Decision Making and Strategic Planning- A Field Survey.
- Ramadge, P.J. & Wonham, W. M. (1989). The control of discrete event systems. *Proceedings of the IEEE*, 77(1).
- Sridhar, B. & Soni, T. & Sheth, K. & Chatterji. G. B. (2006). Aggregate flow model for air-traffic management. *Journal of Guidance, Control, and Dynamics*, 29(4), 992-997.
- Transportation Research Board (2003). Air Transportation Challenges: Airspace, Airports, and Access Aviation. Retrieved October 28, 2010, from <http://www.natca.org>.
- Zhou, M. C. & Venkatesh, K. (1998). Modeling, Simulation and Control of Flexible Manufacturing Systems: A Petri Net Approach. World Scientific, Singapore.

- Zhou, M. C. & DiCesare, F. (1993). *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems*. Kluwer Academic Publishers, London.
- Zhou, M. C. & Li, Z. W. (2009). *Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach*, Springer, New York. 237 pages. *Advances in Industrial Control Series*.
- Zhou, M. C. & Wu, N. Q. (2010). *System Modeling and Control with Resource-Oriented Petri Nets*, CRC Press, New York. 312 pages, *Control Engineering Series*.