A Petri Net Model to Substantiate a Safety-Critical System for a Fail-safe State

B. Esther Sunanda, P. Seetharamaiah
Department of Computer Science and Systems Engineering, Andhra University College of Engineering, (A), Andhra University, Visakhapatnam-530003.

Abstract

Safety critical application industries like automotive, robotics rely on the safety-critical systems whose failure may lead to significant dangers which may be due to the software bugs that may arise in the system. The common methods used by the researchers like Fault Tree Analysis are not well suited due to its ineffectiveness in the issues like interdependencies, data flow. A new framework proposed using Petri Nets called Safety Petri Nets for the development of fail-safe systems is used for developing Petri Net model, reachability graph, reachability set, Safety Petri Nets trees through which the possibility of hazards can be narrowed. For the proposed approach, generation of Safe Failure Fraction and Safety Integrity Levels aid for assuring the echelon of safeness. The corroboration of the proposed approach is done using a laboratory prototype called Four Fingered Robot Hand which is substantiated for the stratum safeness.

Keywords: Safety Petri Net, Safety Petri Net Model, Steady State Probability, Safe Failure Fraction.

I. INTRODUCTION

Safety plays a crucial role in the development of complex computer systems in the relevant areas like automotive, aerospace, power plants, etc. The dramatic change of systems from hardwired to the software oriented computer systems has given an idea that safety need to be included in the development of the systems i.e. it need to be specifically included during the development of the system [2] [3] [1]. The developers of the systems need to induce the safety issues into the system during the software development lifecycle which is unfortunately not done properly. The complex interdependent safety-critical systems due to their large size have made the safety analysis a tough task. There is a requirement of exploiting the software-based critical complex systems for predicting the occurrences of faults [7] [8]. The process of development of the system starts with the analysing of the requirements of the user and developing the design, which is the base of the system.

The software engineers while developing the software use some methods and techniques for the development of the safety-critical systems [3]. The occurrence of accidents did not stop in spite of using regulations in the software development of the safety-critical systems [2]. The failure states of the software may affect the working of the safety-critical system and may lead to catastrophic accidents [3] [6] [8] [10].

The analysis process is not concerned with the software bugs that may arise during the working of the system. However, some specific standards like IEC 61508 can be used for the analysis of the system as well as software issues. Some of the high-profile accidents that occurred in the last decade due to the failure of safety critical systems are listed below which are the eye-opener for a requirement of a specific procedure for the analysis of the system to avoid the occurrences of faults in the system [3] [12] [13][14]:

- On April 9, 2014 the emergency number 911 of United States of America went outage as the security system reached the pre-set limit of the counter 40 million and the system stopped taking emergency calls of about 6,000 in a span of 6 hours.
- In July, 2014 the high speed rail met with an accident in Shanghai, China killed dozens and injured hundreds is due to the failure of the signal lights. The signal lights failed to turn green to red and the reason is due to the lightening strike of the signalling systems which is supposed to be safe even if hit by lightening.
- On August 1, 2012 Knight Capital Group (KCG) Inc. of the US Stock Exchange lost $400 million due to the software error generated in the system which nearly pushed it into bankruptcy.
- In October 2009, the CT scan in Los Angles used for brain scan gave 8 times more radiation to about 200 patients which lead to radiation overdose reporting side effects in the patients is due to the misunderstanding in the embedded default setting in the machine.

In this paper section II deals with the comparative analysis of the safety analytical approaches and section III deals with the proposed safety Petri Nets and section IV deals with the approach for hazard analysis.
II. COMPARATIVE STUDY OF THE SAFETY ANALYSIS APPROACHES

Safety-critical system failures may lead to catastrophic accidents, which are dangerous to the environment and to the people around. The methods used to analyze the failures of Safety-critical systems are Failure Mode and Effects Analysis (FMEA), Failure Modes Effects and Criticality Analysis (FMCEA), Fault Tree Analysis (FTA), Event Tree Analysis (ETA), Failure Mode Factors and Effects Analysis (FMFEA), etc. The features of FMEA and FTA methods are described briefly followed by various analysis techniques like Graphical Requirement Analysis (GRA), Deductive Cause-Consequence Analysis (DCCA) and language like Unified Modelling Language (UML) etc.

A. Failure Mode and Effects Analysis (FMEA)

Failure Mode and Effects Analysis (FMEA) [4] [1] is a method used for analysis to assure quality. It is used for analysing the failures of a system that may lead to hazards. It is used to find their potential failure of a product or a process, to identify and estimate its importance and to recognize appropriate actions to prevent the potential failure of the system. FMEA is used for analysing the individual risks of a system. The risks one-by-one are checked against each other to recognize the failures. FMEA does not provide a report on the total failure risk. For the analysis of failures, fault-tree analysis is more appropriate. The disadvantages of the FMEA technique are they can identify only the major failures of the system and the data flow of the system cannot be represented.

B. Fault Tree Analysis (FTA)

Fault Trees [1] [4] are used for the analysis of a system and to find the probability of failures. A Fault Tree has its representation in the form of a tree. It is used for analysing the failures that may occur due to various conditions. The failure of the system is represented at the top of the tree. A deductive analysis is used to find the failure that is placed at the top of the tree and it provides alternatives for the occurrences of failures. The limitations of FTA are it is a complicated process and the data flow is not represented.

FMEA and FTA have the disadvantages like cannot represent how the data flow occur in a system. Petri Nets are well-suited for the analysis of a system and to identify the failure modes. FTA is constrained to systems whose components have no stochastic inter-dependencies whereas Petri Nets can handle stochastic inter-dependencies in their components. Petri Nets can analyze a system using fail-safe or fault tolerant mechanisms. So, we recommend Petri Nets which are more suitable for analyzing the failures of a system.

C. Event Tree Analysis (ETA)

ETA is an inductive failure analysis performed to determine the consequences of single failure for the overall system risk or reliability [4]. Event Tree Analysis employs related logic as FTA but the process of FTA applies deductive formula whereas ETA employs the inductive formula. An ETA tree is a graphic illustration of introverted failure succession and the sway is on the supplementary systems.

D. Failure Modes Effects and Criticality Analysis (FMCEA)

Failure Modes Effects and Criticality Analysis (FMCEA) is a quality tool which builds on the results of Functional Analysis to identify risks and their consequences [4]. FMCEA can be useful to systems, products, developed procedures, equipment, plants and even low substantive matters such as logistic or information flows. It is employed to recognize the probable ways in which failure can happen for the subsequent causes of malfunction, the equivalent effects of failure, and the bang on purchaser contentment. The purpose of FMCEA is to recognize the mechanism of products and systems nearly all probable to root failure, so that these probable failures can then be intended out. FMCEA permits the recognition early in the product development procedure of possible tribulations or safety hazards which are intrinsic in a product design. The safety and reliability of the product can be calculated and alterations commence at a comparatively squat rate ahead of they are constructed into the product.

E. Failure Mode Factors and Effects Analysis (FMFEA)

FMFEA [4] is a safety analysis method used to analyze factors and effects of failure modes of structural components. This technique employs a blend of FTA and ETA. This method is capable to mutually realize and forecast tribulations by means of an efficient succession of comprehensive analysis. In addition, safety (averting personal damage) and reliability (averting result malfunction) can be examined concurrently.

<table>
<thead>
<tr>
<th>TABLE 1: COMPARISON BETWEEN FMCEA AND FMFEA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FMCEA</strong></td>
</tr>
<tr>
<td>Provides limited quantitative information, or information capable of being measured.</td>
</tr>
<tr>
<td>It is used by the U.S. Army to assess mission critical equipment and systems.</td>
</tr>
<tr>
<td>Analysts must perform FMEA followed by critical analysis (CA).</td>
</tr>
<tr>
<td>Attaches a level of criticality to failure modes.</td>
</tr>
<tr>
<td>Takes up a single failure mode with a basic and proper function obliterated.</td>
</tr>
</tbody>
</table>
TABLE 2: COMPARISON BETWEEN FMEA, FTA AND ETA

<table>
<thead>
<tr>
<th>FMEA</th>
<th>FTA</th>
<th>ETA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides only qualitative information but not quantitative information.</td>
<td>Focuses on a single failure mode and identifies all of the failure mechanisms.</td>
<td>Analysis performed to determine the consequences of single failure for the overall system risk or reliability.</td>
</tr>
<tr>
<td>FMEA is used to analyze reliability.</td>
<td>FTA is used to analyze reliability, safety.</td>
<td>ETA is used to analyze reliability and safety.</td>
</tr>
<tr>
<td>Analysis is from Component failure mode.</td>
<td>Analysis is from Product failure, injury.</td>
<td>Analysis is from Component failure mode.</td>
</tr>
<tr>
<td>Components→ Product (Bottom up)</td>
<td>Product→ Components (Top down)</td>
<td>Components→ Product (Bottom up)</td>
</tr>
<tr>
<td>Qualitative analysis is done.</td>
<td>Both quantitative and qualitative analysis is done.</td>
<td>Both quantitative and qualitative analysis is done.</td>
</tr>
<tr>
<td>FMEA does the advance analysis.</td>
<td>Both advanced and after fact analysis are done.</td>
<td>Both advanced and after fact analysis are done.</td>
</tr>
<tr>
<td>FMEA is represented by FMEA table.</td>
<td>FT is represented by FT diagram.</td>
<td>ET is represented by ET diagram.</td>
</tr>
<tr>
<td>Applicable components are selected from function block diagram.</td>
<td>Range of analysis is limited to top event (for analysis target)</td>
<td>Range of analysis is limited to initial event (for analysis target)</td>
</tr>
<tr>
<td>Insufficient effects analysis (limited to single failure).</td>
<td>Cause analysis done logically and systematically.</td>
<td>Effect analysis done logically and systematically.</td>
</tr>
<tr>
<td>No misuse at analysis of safety.</td>
<td>No misuse at analysis of safety.</td>
<td>Applicable to misuse at initial event (limited)</td>
</tr>
</tbody>
</table>

The above specified tables 1 and 2 demonstrate the comparative study of the analytical techniques but fail to specify the properties of stochastic interdependencies and data flow which are the main things to accomplish safety to the system.

F. Other Analysis Techniques

Graphical Requirement Analysis (GRA) generates logic bases graphical representation from functional requirements. Functional requirements are recognized by the GRA but are not in detail, so they are not used for the safety analysis. Deductive Cause-Consequence Analysis (DCCA) takes the mathematical procedures to recognize the failures of a system. DCCA uses CTL and here the CTL uses the time requirements based on branching and FTA uses the time requirements based on linearity. UML can be used for the analysis of the system and can be combined with other methods. Petri Nets use backward analysis to study hazards and determine critical areas of the system and this helps to use fault tolerant or fail-safe safety mechanisms to address the critical areas of the system.

III. SAFETY PETRI NETS

Petri Nets can analyze a system using fail-safe or fault tolerant mechanisms as they can specify the dataflow and use the stochastic interdependencies for identifying the failure modes of a system [10][25]. For the analysis of any system, Petri Nets are better able to recognize hazards when compared to FTA and FMEA. So, we recommend Petri Nets, which are more suitable for analyzing the failures of a system. Data Flow representation is not represented in FTA, which is much needed for analysis of the hazards and Petri Nets helps the designer to use fault tolerant or fail-safe safety mechanisms to analyze the failures of a safety-critical system. In SPN, immediate transitions, priority transitions and probability are considered separately unlike GSPN. Here, we propose a eleven-tuple Safety Petri Nets notation which is specified using the principle of Generalised Stochastic Petri Nets (GSPN) i.e. SPN=(P, T, F, π, I, O, H, M0, W, Pr,T) where P is the set of places i.e. P = {P1, P2, P3 ......}; T is the set of transitions i.e. T= {T1, T2, T3......}, T ∩ P = ø; F: (P X T) U (T X P) is a finite set of arcs; π: T→ N is the priority function that maps transitions onto natural numbers representing their priority levels where N is a set of non-negative integers; I: P X T → N is an input function that has directed arcs from places to transitions; O: T X P → N is an input function that has directed arcs from transitions to places; H: T → P is an input function that has directed arcs from transitions to places that take no time to fire; M0: P → N is the initial marking; W: P X T → N is a non-negative integer assigned to each transition to specify the delay of fire; Pr: Pr → N is a value associated with the Probability of firing of events;
\( \tau : T \rightarrow N \) is a function that associates transitions with deterministic time delays;

**Definition 1:**
A place \( p \in P \) is said to be **safe** in a Petri Net at marking \( M \) if every marking \( M_1 \) reachable from \( M \) has at most one token on \( p \) i.e. \( M_1 \geq 1 \).

**Hypothesis:**
- A Petri Net is faulty if token \( n \) is struck in the place \( p \in P \).
- A Petri Net model is fault-free, if it is loop-free and if \( I \neq \phi \) and \( O \neq \phi \). The transition need to fire immediately if it is more prioritized and need to consider the deterministic time delays.

**Lemma 1:**
A place \( p \in P \) where \( I \neq \phi \) and \( O \neq \phi \) is safe if \( t \in T \) fires to \( p^1 \) for every initial marking for a fault-free graph of SPN.

**Proof:**
Let \( p \) be a place of a fault-free SPN model and consider it as safe for an initial marking. If \( p \) is not safe then there exists fault SPN model for an initial marking \( M \). A SPN is safe for a marking \( M \) if every place of the SPN is safe at the marking \( M \) and let SPN be loop free. If in the fault SPN model a token is fired from the initial marking \( M^0 \) then there exists a path from \( M^0 \) to \( M^* \) with a firing sequence \( \eta \) then \( M^0 (p) \geq 2 \) and if in the fault SPN model a token is fired from the initial marking \( M^0 \) then there exists a path from \( M^0 \) to \( M^0 \) without a transition \( t \) then \( M^0 (p) \geq 2 \) which is a contradiction. (OR)

Let \( p \in P \) be the place of a fault-free SPN then if a token is fired from \( p \rightarrow p^1 \) then \( t \in T \) need to included with:

1) Each \( t \in T \) needs to be included with a time delay \( \tau \) for stopping the occurrence of hazards.
2) Each \( t \in T \) needs to be included with a \( \pi \) for the firing of a token with priority.
3) Each \( t \in T \) needs to be included with \( H \) for the firing of a token immediately.

Then a token can fired from \( p \rightarrow p^1 \) with \( t \in T \) and a fault-free SPN can be generated.

**IV. FRAMEWORK FOR SUBSTANTIATING A SAFETY CRITICAL SYSTEM**

The proposed framework for the hazard analysis of the safety-critical system is for observing the faults [7] [8] [9] [10] of the safety-critical system and specifies a procedure for the analysis of the occurrence of failure modes as shown in figure 1. For a system while being developed, the liable data set \( (\lambda) \) is considered which has the details of the places where there is a chance of occurrence of accidents. The liable dataset contains the model fault details which once resolved can make the safety-critical system work effectively. The liable dataset is given as the input for the SPN model which is developed using the proposed SPN where the things like priority, immediate transitions, fault nodes, safe nodes, etc., are considered. The SPN model helps to cluster the data flow activity in such a way that the path represents a way to the safe state. The SPN model is developed using the Petri Net notations. The proposed framework for hazard analysis takes the input liable dataset processes it and gives the Steady State Probability [12] and Safe Failure Fraction through which again the liable dataset is generated as input for the SPN model for further processing. The working of the framework for hazard analysis is cyclic as no safety-critical system is proven to be genuine 100%.
V. SPN ANALYSIS OF FOUR FINGERED ROBOT HAND: A CASE STUDY

The proposed SPN framework is experimented on the laboratory prototype Four Fingered Robot Hand (FFRH) for the safety analysis of the system [15]. The FFRH system as shown in figure 2 has three main parts namely Robot Hand, Controller and Sensor Feedback unit. The Robot Hand consists of three fingers and an opposing thumb and is connected internally to the geared DC motors, for their movement. The FFRH controller is based on embedded processor, relay-based geared dc-motor drivers, LCD display for messages and debugging, an in-system programming (ISP) port for downloading programs from a PC to embedded processor’s flash memory, data from IR sensors and limit sensors of hand via another embedded processor based encoder to control the limits of movement of hand, 5x5 key matrix interface using parallel port for hand’s control commands [15].

All the joints of the hand including the base are controlled by geared DC motors using strong gut type wires wound from the shaft to the finger joint pulleys and idlers and relay drivers interfaced to parallel port bits of embedded processor [15]. Base motor is used to move hand in 360 degrees in either direction programmed by key commands (base-cw, base-ccw). Four fingers use four geared DC motors/drivers with relays, for motion control of finger joints with pitch of 45 degrees maximum. Wrist geared DC motor drives the wrist with roll and pitch of maximum 180 degrees. The input from finger limit sensors placed on palm is used to control the free rotation of the finger. IR sensor placed at the center of the Palm is used to sense the presence of an object. All operations of embedded computer-based robot hand are achieved through control program [15].

A. Obligation of Safety Issues

The safety issues need to be included in the design phase of the Four fingered Robot which is not included in the actual design [15]. The inclusion of the Safety Issues in the design phase should be present as without it accidents can occur in the Four Fingered Robot Hand. The sensors in the Four Fingered Robot are four individual sensors for each finger to move to and fro, three sensors in the wrist to move it and a sensor in the base which is used for the purpose of keeping the base in the home position [15].

When the finger of the robot moves up the sensor present at the back of the finger detects its position i.e. whether the finger is up or down. At the wrist there are three sensors which specify the correct position of it. There is a sensor at the base which specifies whether the Four Fingered Hand is in its home position or not. There may be a case when the sensors in the above cases may not work properly, then the hand will not properly work. The objects that are to be hold by the hand may fall off or other damages may occur.
In the figure 3, the operation of the Robot Hand has been specified where there are no specifications regarding the safety issues which can be included using Safety Petri Nets (SPN). Initially, using the SPN representation \( \text{SPN} = (P, T, F, \pi, I, O, H, M_0, W, \text{Pr}) \) a Safety Petri Net model is developed where the inclusion of safety issues in the model are considered like stack overflow, masking of bits, etc., Based on the model a reachability set and a reachability graph are developed which pave a path for the development of SPN Trees which aid for the recognition of faults that may occur in the system. Based on the information of SPN Trees, the designed system need to be re-designed if required and the safe code need to be included for the safe working of the system.

The generation of errors in the Four Fingered Robot Hand (FFRH) may occur during its working i.e. when the software is executed [15]. The errors that may occur during its execution can prove more dangerous or can be less dangerous based on their occurrence. The errors that are considered during its working are stack errors, initialization errors, masking of bits, call of subroutines, call for delay, etc., these are specifically shown in the diagram of the Petri Net model which needs to be avoided for the smooth working of the FFRH. The places of error generation are marked as the zone of occurrence of faults where extra precautions need to be taken. Table 3 shows the list of elements like places and transitions used in the model.

### TABLE 3: LIST OF ELEMENTS USED IN THE SAFETY PETRI NET MODEL OF FFRH

<table>
<thead>
<tr>
<th>P1</th>
<th>Initialization of the system</th>
<th>P38</th>
<th>Mask all bits except the 5th bit which represents the down position of the wrist</th>
</tr>
</thead>
<tbody>
<tr>
<td>P2</td>
<td>Data accessed by the Carbon brushes</td>
<td>P39</td>
<td>Make the corresponding C1 bit low in control port to stop taking down the wrist</td>
</tr>
<tr>
<td>P3</td>
<td>Data accessed by the Copper tracks</td>
<td>P40</td>
<td>System halts if the masking of the bits not done properly</td>
</tr>
<tr>
<td>P4</td>
<td>Initialization of the Micro-controller</td>
<td>P41</td>
<td>System halts if the masking of the bits not done properly</td>
</tr>
<tr>
<td>P5</td>
<td>Initialization of the Stack Pointer value</td>
<td>P42</td>
<td>System reaches the safe state</td>
</tr>
<tr>
<td>P6</td>
<td>Motors in off-mode</td>
<td>T1</td>
<td>Receives the initialized data which is fired immediately</td>
</tr>
<tr>
<td>P7</td>
<td>Initialization of the Ports</td>
<td>T2</td>
<td>Gets the data regarding the initialization of the hardware</td>
</tr>
<tr>
<td>P8</td>
<td>Display the WELCOME message on the LCD display</td>
<td>T3</td>
<td>The data regarding the micro-controller is obtained.</td>
</tr>
<tr>
<td>P9</td>
<td>System halts if the stack error occurs</td>
<td>T4</td>
<td>Receives the data regarding the stack</td>
</tr>
<tr>
<td>P10</td>
<td>System halts if the initialization of the ports not done properly</td>
<td>T5</td>
<td>Holds the data regarding the motors</td>
</tr>
<tr>
<td>P11</td>
<td>Selection of the key</td>
<td>T6</td>
<td>Gets the data regarding the ports</td>
</tr>
<tr>
<td>P12</td>
<td>Selection of the finger for the open or close mode</td>
<td>T7</td>
<td>Takes the information about the LCD display</td>
</tr>
<tr>
<td>P13</td>
<td>Masking of the bits for the position of the finger in open mode</td>
<td>T8</td>
<td>Collects the information regarding the key</td>
</tr>
<tr>
<td>P14</td>
<td>Selection of the wrist for the up or down mode</td>
<td>T9</td>
<td>Enters the data regarding the data port</td>
</tr>
<tr>
<td>P15</td>
<td>Masking of the bits for the position of the finger in close mode</td>
<td>T10</td>
<td>Enters the data regarding the control port</td>
</tr>
<tr>
<td>P16</td>
<td>Make the bit high for the port and call the subroutine</td>
<td>T11</td>
<td>Receives the data regarding the mode</td>
</tr>
<tr>
<td>P17</td>
<td>Make the bit low for the finger to stop closing</td>
<td>T12</td>
<td>Receives the data regarding the selection mode</td>
</tr>
<tr>
<td>P18</td>
<td>Call the subroutine to read the serial port</td>
<td>T13</td>
<td>Helps to select the rotation modes</td>
</tr>
<tr>
<td>P19</td>
<td>Mask all bits except the bit which represents the position of the finger</td>
<td>T14</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P20</td>
<td>Make the corresponding bit low in data port to stop opening the finger</td>
<td>T15</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P21</td>
<td>System halts if the data port bit is not made low</td>
<td>T16</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P22</td>
<td>Make the corresponding bit high in data port to open the finger</td>
<td>T17</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P23</td>
<td>System halts if the masking of the bits are not done properly</td>
<td>T18</td>
<td>Helps to select the clock wise mode</td>
</tr>
<tr>
<td>P24</td>
<td>Call the subroutine to read the current position of the hand and mask all bits except the 6th bit which represents the up position of the wrist</td>
<td>T19</td>
<td>Helps to select the counter clock wise mode</td>
</tr>
<tr>
<td>P25</td>
<td>Call the subroutine to read the current position of the hand and mask all bits except the 5th bit which represents the up position of the wrist</td>
<td>T20</td>
<td>Get the data for port</td>
</tr>
<tr>
<td>P26</td>
<td>System halts if the masking of the bits not done properly</td>
<td>T21</td>
<td>Get the data for port</td>
</tr>
<tr>
<td>P27</td>
<td>Make the C7 bit high in control port to raise the wrist, call the subroutine to read the position of hand and mask all bits except the 6th bit for the up position of the wrist</td>
<td>T22</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P28</td>
<td>Make the C1 bit high in control port to take the wrist down</td>
<td>T23</td>
<td>Get the data for control port</td>
</tr>
<tr>
<td>P29</td>
<td>System halts if the masking of the bits not done properly</td>
<td>T24</td>
<td>Helps to call the subroutine</td>
</tr>
<tr>
<td>P30</td>
<td>Selection of the base for base clock wise or counter clock wise mode</td>
<td>T25</td>
<td>Helps to call the subroutine</td>
</tr>
<tr>
<td>P31</td>
<td>Selection of the base for base clock wise mode</td>
<td>T26</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P32</td>
<td>Selection of the base for base counter clock wise mode</td>
<td>T27</td>
<td>Accumulates the data for bit masking</td>
</tr>
<tr>
<td>P33</td>
<td>Make the corresponding C8 bit high in control port to rotate base in anticlockwise direction</td>
<td>T28</td>
<td>Gets the data low for the finger to close</td>
</tr>
<tr>
<td>P34</td>
<td>System halts if the selection mode is not proper</td>
<td>T29</td>
<td>Get the data for control port</td>
</tr>
<tr>
<td>P35</td>
<td>Make the corresponding C1 bit high in control port to rotate base in clockwise direction</td>
<td>T30</td>
<td>Get the data for control port</td>
</tr>
<tr>
<td>P36</td>
<td>Call the subroutine to read the position of hand through serial port</td>
<td>T31</td>
<td>Get the data for control port</td>
</tr>
<tr>
<td>P37</td>
<td>Make the C2 bit low in control port to stop raising the wrist</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
B. Hazard Analysis of the FFRH

Step 1: Development of Safety Petri Net Model of the FFRH

FIGURE 4: SAFETY PETRI NET MODEL FOR FFRH

Step 2: Generation of Reachability Set

\{ (1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0),
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0),
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0),
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0),
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0),
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1) \}
Step 3: Generation of Reachability Graph

FIGURE 5: REACHABILITY GRAPH FOR FFRH

Step 4: Construction of SPN Trees

FIGURE 6: SPN TREE FOR FFRH
Step 5: Analyze Relation between Errors

As stated in step 1 the developed Petri Net model halts when it reaches the states P9, P10, P41, P23, P21, P26, P29, P40, P34 and it is due to the occurrence of errors in the Four Fingered Robot Hand (FFRH) software. At the places of occurrence of errors, extra precautions need be taken for avoiding accidents. The places of the Petri-Net model where the accidents are more likely to occur are represented with a priority node and the places where the transitions have to be fired immediately are represented with an immediate transition. At the places where there is a chance of occurrence of failure a delay in the code can be included by which if the action is not completed the system need to halt to avoid the happening of accidents.

Step 6: Rectified Design

Based on the developed Petri Net model, Reachability Set and Reachability Graph the errors that may occur during the development of FFRH can be recognized easily. The recognized errors can be solved and can be rectified in the design phase of the system. The places where errors may occur extra precautions are taken like they are considered as prioritized nodes and sometimes the data of it is immediately transferred. Using these precautions, the Petri Net tree is developed which is error free and using this design can be modified.

C. Safe Code

Based on the developed Petri Net model, the places where there is a chance of hazards are recognised. The code regarding those places is modified to make it safe. The time factor of the SPN tuple format is included in the software part of the program using timers, counters, etc., There may be a case where the sensors in the FFRH may not work properly which may lead to the faulty working of the hand due to which the objects that are held by the hand may fall off or other damages may occur [15]. Therefore, the software of the Four Fingered Hand must include a clause which checks for the proper working of the FFRH. If the sensors present there work correctly, the operations of the wrist, fingers and the base can be monitored with the safe code. The software code before the inclusion of safe code of the Four Fingered Robot Hand for a single finger to operate in the OPEN phase is:

1) BEGIN.
2) Call the subroutine to read the current position of the hand through serial port.
3) Mask all bits except the bit which represents the position of the finger.
4) If the result is equal to zero then go to step 9 else go to step 5.
5) Make the corresponding bit high in data port to open the finger.
6) Call the subroutine to read the serial port.
7) Mask all bits except the bit which represents the position of the finger.
8) Return result to step 4
9) Make the corresponding bit low in data port to stop opening the finger.
10) END.

Here in the above code, if the sensor behind the finger is not in the working condition then it may not proceed with the next operation of CLOSE and the code moves to an undefined loop. So, there must be a safe code which must be included so that the OPEN state of the Four Fingered Robot when the sensor is in the non-working condition can be done without any interruption.

1) BEGIN.
2) Initialize a counter and compare with the pointer value if equal goto step 3 else goto step 11. (Safe Code)
3) Call the subroutine to read the current position of the hand through serial port.
4) Increment the value of counter. (Safe Code)
5) Mask all bits except the bit which represents the position of the finger.
6) If the result is equal to zero then go to step 9 else go to step 5.
7) Make the corresponding bit high in data port to open the finger.
8) Call the subroutine to read the serial port.
9) Mask all bits except the bit which represents the position of the finger.
10) Return result to step 4
11) Make the corresponding bit low in data port to stop opening the finger.
12) END

These extra lines of safe code of step2 and step4 need to be included so that safety can be provided to the Robot Hand while the sensors are not in the working condition. The code described here is for one finger but there are 8 sensors and 3 fingers in the Robot Hand which need to be checked [15]. The safe code need to be applicable for the checking the operation of every sensor i.e. for the 4 sensors of fingers, the 3 sensors of the wrist and the one sensor of the base [15].

D. Steady State Probability and Safe Failure Fraction (SFF)

Definition 1: A Steady State Probability distribution \( \pi \) is defined by the fact that if the distribution of the system is \( \pi \) at a given time, it is still \( \pi \) at any later time.

Definition 2: Safe Failure Fraction (SFF) is the ratio of the safe failure rate of a subsystem plus the dangerous detected failure rate of the subsystem) to the total failure rate of the subsystem i.e. it specifies how fail-safe the system is. The formula used for the calculation of SFF is [16]:
The generation of the Steady State Probability and Safe Failure Fraction helps to prove the percentage of safeness of the proposed safety-critical system. The Steady State Probability gives the percentage of safeness in the safety-critical system. Based on the outcome i.e. the percentage of safeness of the safety-critical system SFF is calculated using the above formula. Using the generated SFF, the SIL to which the safety-critical system belongs to is estimated based on whichever modifications to the safety-critical system is needed are done to upgrade the safeness of the safety-critical system via the levels of table 3. In the table 3, safety is divided into 4 levels and each level describes the extent of unsafe reduction. The higher the SIL level the lower is the failure rate that is acceptable. Hardware Fault Tolerance is the ability of the hardware (hardware and software) to persist to carry out a required operation in the existence of errors or faults. A hardware fault tolerance of 0 means that if there is one fault the system will not be able to perform its function. A hardware fault tolerance of K means that (K+1) faults could cause a loss of the safety function.

VI. CONCLUSION

The Safety Petri Net Framework helps to build the safety critical systems in a safe, fault-free and riskless manner. The hazard analysis of the software-based safety critical system is done using the 11-tuple Safety Petri Net which has been used for the development of Safety Petri Net model where the occurrence of errors in the system are recognized with the reachability set, reachability graph for analyzing the errors. The proposed framework for hazard analysis helps to figure-out the Safety Integrity Level which aids to find the safeness of the safety-critical system and is experimented on a robot hand called the Four Fingered Robot Hand where the results were found satisfactory. The approach can further be extended with the calculation of weights of the Petri Nets which are useful for the calculation of safeness and reliability of the safety-critical system.

REFERENCES


