CAR'S OBJECT DETECTION AND AVOIDANCE SIMULATION USING VERILOG

By

MOHD KHUWARIZMI BIN MOHAMAD ROSLAN
4104

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

CAR'S OBJECT DETECTION AND AVOIDANCE SIMULATION USING VERILOG

by

Mohd Khuwarizmi Bin Mohamad Roslan

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

Ms Salina Bte Mohamad
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

June or December 2007

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

Mohd Khuwarizmi Bin Mohamad Roslan

ABSTRACT

The advancement of automotive industry in the last century has produced better and faster car. However, all that comes with worrying trend of increasing accident on the road. A key challenge today is to develop electronics gadgets or equipment that can help to reduce the statistic significantly. Advanced safety systems that sense impending danger – such as a collision or a potential rollover – and alert the driver can be introduced on the vehicle to achieve the target of reducing accidents. Multiple high-end sensors can be used to diagnose the car conditions or to detect nearby objects and irregularities. On-board processing will then be performed to determine the appropriate actions to be taken. The processor will then communicate with one of the actuators in view of alerting the driver or avoiding the danger. This project aims to simulate the necessary action to be taken when the car is within the danger range. For this project, the scope of the accident causality is only focus on crushing the object/vehicle in front and the necessary action taken are to be the driver alert system and engaging Anti-lock Brake System (ABS).

ACKNOWLEDGEMENTS

First and foremost, thank to Allah swt , the most Gracious & the most Merciful. Much gratitude to Ms Salina Mohamad for her guidance and advice through out the project development. Thanks to my friends who help me along the way in the projects.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CHAPTER 1
INTRODUCTION

1.1 Background

Car accident's rate has been increasing from time to time. This results in severe injuries and deaths. As vehicle are generally traveling faster and the roads are getting more crowded than before, driving has become more difficult than before. Many reasons have been associated with the accidents, but by being extra alert to their driving, a driver can avoid the imminent collision.

A Car's Object Detection and Avoidance Simulation using Verilog (CODA) is a simulation of a system that helps the driver to be alerted to the incoming danger or even takes necessary action automatically. The system consists of a list of sensors, a central electronics system and some actuators. An accelerometer is being used to determine the running state of the vehicle.

Other sensors such as front detector is added to detect the nearby objects and together with the car running data, the intelligent electronics system can determine whether the driver is in danger or not. The driver will be alerted first once the system triggered the distance between car and the object is within the danger zone. If there are no responses taken from the driver, the system will automatically execute the Anti Lock Brake System (ABS). The force to be applied to the ABS depends on the distance between the car and the object. The shorter the distance the more force will be applied.

1.2 Problem Statement

Many alternatives and methods have been done to reduce the accident rates in Malaysia. However, there is still no positive result and yet the accident rate is increasing. Base on the analysis done by the Ministry of Transportation, the main cause of the accident is the drivers themselves; the attitude of driving, carelessness and others. On the other hand, there are also factors that can cause the driver to lost control of the vehicles such as road and the vehicle conditions.

Even though the car's manufacturers nowadays have applied the advanced technology in order to increase the safety features of their cars, accident rate is still at the critical level. Obviously, it is not what the technology applied that affluence the accident to occur, it is a matter of how the technology and the safety features are used. In other words, it is the responsibility of the drivers to ensure their safety. However, human are not machines. The tendency of being fatigue, less focus or panic can cause them not to use the technologies or safety features provided when they are supposed to use them.

Accidents can occur in milliseconds time and under critical time human normally cannot think and react rationally. This is where an intelligent system can 'assist' the driver. They can 'think' for the driver and even 'react' if the driver's response is slow. The system however has to be very accurate in order to ensure the best corrective action is taken to avoid any accident.

1.3 Objectives and Scope of Study

The main goal of this project is to simulate a system of Car's Object Detection and Avoidance Simulation using Verilog. The system is capable of determining an event where a car approaches an object or vice a versa. By knowing the approaching speed and also the car speed, the circuit shall be able to determine the collision time and take necessary actions such as warn the driver and engaging Anti-lock Braking System (ABS). In developing the simulation, the Quartus Version 6 is used.

CHAPTER 2
LITERATURE REVIEW

2.1 Increase of Research in Improving Vehicle Safety Features.

For the past few years, automotive industries are more focus on producing powerful engine that can increase the power and the speed of a car. Many researches and studies have been done in order to achieve their vision. However, they are neglecting the side effect of having a greater engine in a car, which is the safety. Providing the powerful engine in the car not only increase the speed of a car but also increase the possibility of having a collision or accident. As a result of the 'imbalance' automotive improvement, the accidents rate is increase globally. Nowadays, car manufacturers have realize this issue and have played their parts in overcome this problem. New cars not only have greater power but also better safety features. For instance, the invention on auto cruise control, airbag system, Anti-lock Braking System (ABS) and money others. Nevertheless, the accident rate does not decrease as expected and yet, it is still increasing from year to year. Studies have been done on this matter and it suggest that providing the driver with as many safety features as possible does not ensure the accident avoidance unless the technologies are used. Therefore, an intelligent system which can integrate all the technologies and 'communicate' with the driver is a best solution in this matter. Lately, many parties have joined the effort of inventing such system. The next page shows the examples of the studies and invented intelligent system. [1] [2]

2.1.1 e-Safety System [3]

*e*Safety systems - formally called Intelligent Vehicle Safety Systems (IVSS) - are new automotive systems combining mechanical, microelectric, communication and information technology. They create superior safety through active technology. *e*Safety systems contribute to safety on roads by preventing vehicle collisions and consequently helping to reduce injuries and deaths on the roads.

Below you can find information regarding a number of *e*Safety systems - vehicle based and infrastructure based systems. These can be divided into active and passive safety applications. Passive safety features help people stay alive and uninjured in a crash, while active safety features help drivers avoid accidents.

2.1.1.1 Adaptive Brake Lights

Triggered by the strengths of brake activation the rear brake lights are illuminated in different kinds to indicate emergency braking maneuvers to the following vehicles.

2.1.1.2 Automatic Headlight Activation

When activated, the system switches on the headlights automatically when major environmental conditions for the use of headlights are present. The system detects the darkness and the light conditions in the environment.

2.1.1.3 Driver Condition Monitoring

The system monitors the condition of the driver. Discussed parameters today are drowsiness, distraction, and inattention.

2.1.1.4 Dynamic Control Systems

Active Front Steering: The AFS allows - electronically controlled – a variable steering transmission and steering force support. Two different inputs overlap, the steering angle from the steering wheel and a correction angle given by a controller through a special gearbox.

Electronic Stability Control (ESC): Stabilises the vehicle under all driving conditions and driving situations within the physical limits. Helps to stabilise the vehicle and prevent skidding when cornering or driving off through active brake intervention on one or more wheels and intelligent engine torque management.

Active Body Control (ABC): Active damping and suspension system minimising car body roll and pitch motion, adjusting ground clearance according to speed, allowing for a two stage ride height including load-independent all-round self-levelling.

2.1.1.5 Lane Departure Warning

Warning given to the driver in order to avoid leaving the lane unintentionally. Video image processing is the most important technology.

2.1.1.6 Obstacle& Collision Warning

System detects obstacles and gives warnings when collision is imminent. Current solutions with limited performance are a separate feature of Adaptive Cruise Control systems, which use information obtained from radar sensors to give visual and acoustic warnings. Future systems will use long range/near range radar sensors or LIDAR and video image processing.

2.1.2 Cooperative Intersection Collision Avoidance Systems (CICAS) [4]

In 2003, more than 9,000 Americans died and roughly 1.5 million Americans were injured in intersection related crashes. Intersection collision avoidance systems can help save lives by preventing these crashes. Through the Cooperative Intersection Collision Avoidance Systems initiative, the USDOT is working in partnership with the automotive manufacturers and State and local departments of transportation to pursue an optimized combination of autonomous-vehicle, autonomous-infrastructure and cooperative communication systems that potentially address the full set of intersection crash problems.

2.1.2.1 CICAS Overview

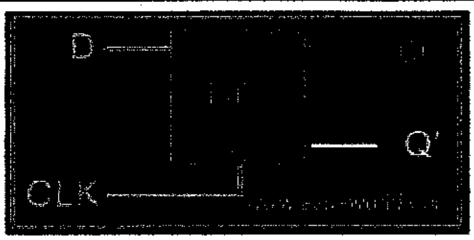
Intelligent intersection systems offer a significant opportunity to improve safety by enhancing driver decision-making at intersections that will help drivers avoid crashes. Intersection collision avoidance systems use both vehicle-based and infrastructure-based technologies to help drivers approaching an intersection understand the state of activities within that intersection. Cooperative intersection collision avoidance systems (CICAS) have the potential to warn drivers about likely violations of traffic control devices and to help them maneuver through cross traffic. Eventually, CICAS may also inform other drivers (i.e., potential victims) about impending violations as well as identify pedestrians and cyclists within an intersection.

CICAS consists of:

- Vehicle-based technologies and systems—sensors, processors, and driver interfaces within each vehicle.
- Infrastructure-based technologies and systems—roadside sensors and processors to detect vehicles and identify hazards and signal systems, messaging signs, and/or other interfaces to communicate various warnings to drivers.
- Communications systems—dedicated short-range communications (DSRC) to communicate warnings and data between the infrastructure and equipped vehicles.

2.2 Programming Language Used for the Project [5]

For this project, the language used is Verilog. Verilog is a hardware description language (HDL) used to describe a digital system: for example, a network switches, a microprocessor or a memory or a simple flip-flop. It means that, by using a HDL, one can describe any digital hardware at any level.




```
// D flip-flop Code
module d_ff ( d, clk, q, q_bar);
input d ,clk;
output q, q_bar;
wire d ,clk;
reg q, q_bar;

always @ (posedge clk)
begin
  q <= d;
  q_bar <=  ! d;
end

endmodule
```

Figure 1 : Example of Verilog coding

One can describe a simple flip flop as that in Figure 1, as well as a complicated design having one million gates. Verilog is one of the HDL languages available in the industry for hardware designing. It allows us to design a digital design at Behavioral Level, Register Transfer Level (RTL), gate level and switch level. Verilog allows hardware designers to express their designs with behavioral constructs, deferring the details of implementation to a later stage in the final design.

2.2.1 Design Styles

Verilog, like any other hardware description language, permits a design in either Bottom-up or Top-down approach.

2.2.1.1 Bottom-Up Design

The traditional method of electronic design is bottom-up. Each design is performed at the gate-level using the standard gates. With the increasing complexity of new designs this approach is nearly impossible to maintain. New systems consist of ASIC or microprocessors with a complexity of thousands of transistors. These traditional bottom-up designs have to give way to new structural, hierarchical design methods. Without these new practices it would be impossible to handle the new complexity.

2.2.1.2 Top-Down Design

The desired design-style of all designers is the top-down approach. A real top-down design allows early testing, easy change of different technologies, a structured system design and offers many other advantages. But it is very difficult to follow a pure top-down design. Due to this fact most designs are a mix of both methods, implementing some key elements of both design styles.

CHAPTER 3

METHODOLOGY

3.1 Introduction

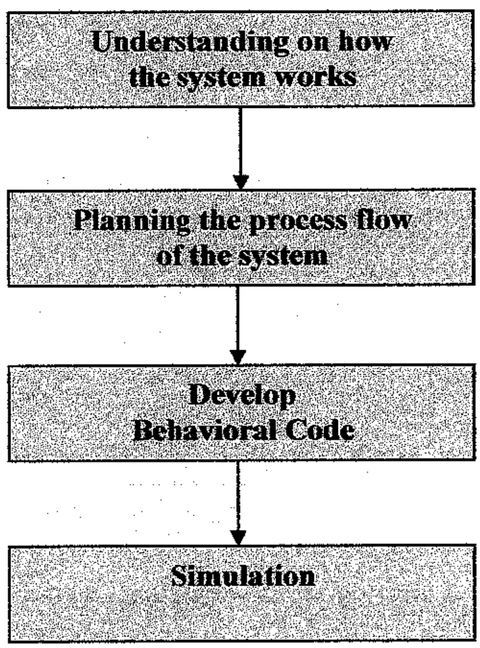


Figure 2 : Process flow of the project

Figure above shows the entire process flow of the project. There are four major steps in completing this project. The first two steps is generally the understanding and designing of the system. These steps are crucial since they will determine the type of the system. Thorough understanding and analysis is necessary. For the last two steps, they are more on the system development; develop the coding and simulation of the coding. On the next page, each of the steps will be discussed in detail.

3.1.1 Understanding on how the system works

Understanding the whole system is the fundamental of this project. The simulation of the system should base on the real and actual conditions so that it can be used as a guide or reference for future development of the system. The understanding includes the location of the sensors, data processing and concept of physics.

3.1.2 Planning the process flow of the system

Upon understanding how the system works, the process flow of the system is determined based on the specification and requirements need. The flow includes all the process from the beginning until the end considering all possibilities.

3.1.3 Develop behavioral code

From the process flow, the module of the system is determined in order to develop the behavioral code. Dividing the system into different module will make the system easier to develop/trouble shoot or to be improved in the future.

3.1.4 Simulation

Completing the behavioral code, the simulation of the system can be done. The results from the simulation will be analyzed and correction or improvement can be made should there are any changes required.

CHAPTER 4
RESULT AND DISCUSSION

4.1 Understanding on how the system works

Figure 3 shows the probable locations of the sensors and actuators. *Figure 4* shows the connections between them and the components of the Intelligence Processing Unit (IPU). Most of the times the data produced or accepted by each of the sensors and actuators are not in the same format or standard. The data can be in analog format (voltage or current levels) or even in digital format. That is why the sensors are being connected to the processor's Data Acquisition Module (DAM) that will process the data before it can be used in data processing.

IPU will accept data from the sensor through DAM and process them immediately. IPU is programmed in a loop that processes all the sensors' inputs and translates them into the vehicle running state. The state will then be compared to safe condition data stored in the IPU. Any irregularities will trigger the driver's alarm system. Should there are no response taken by the driver, the IPU will then execute the Antilock Brake System (ABS) automatically.

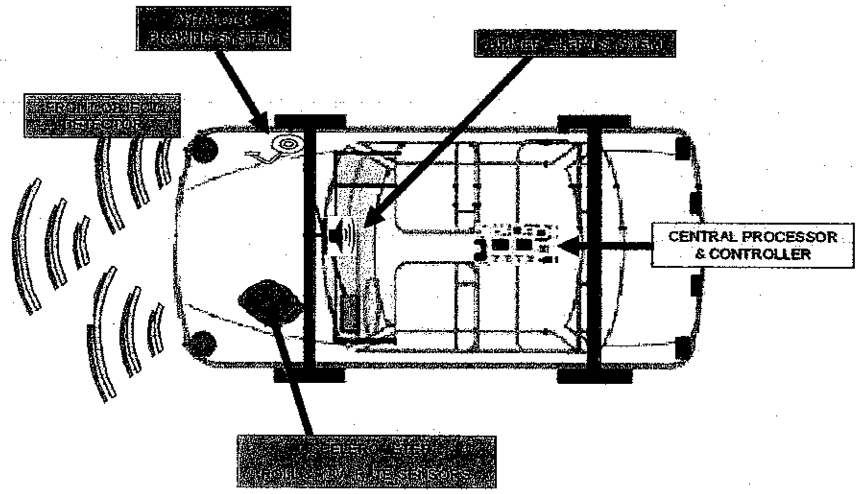


Figure 3 : Conceptual diagram

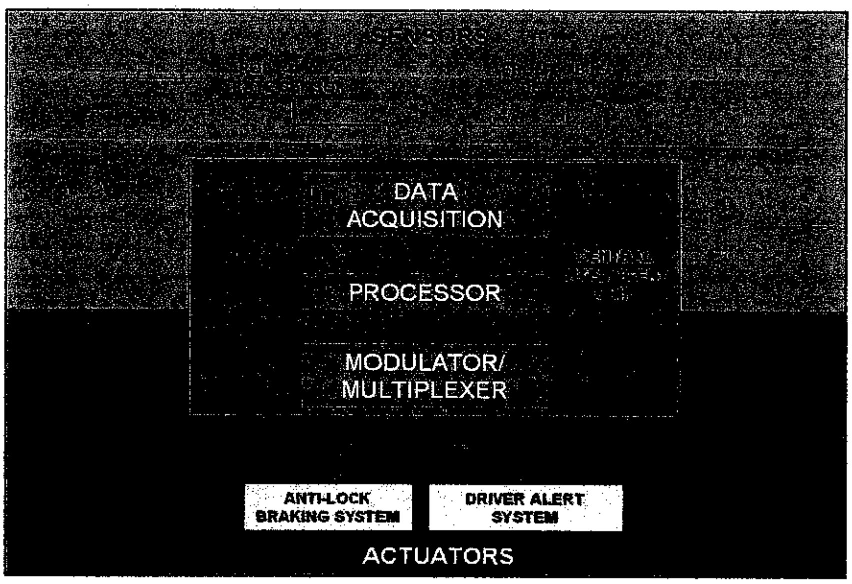


Figure 4 : System block diagram

4.2 Process flow of the system

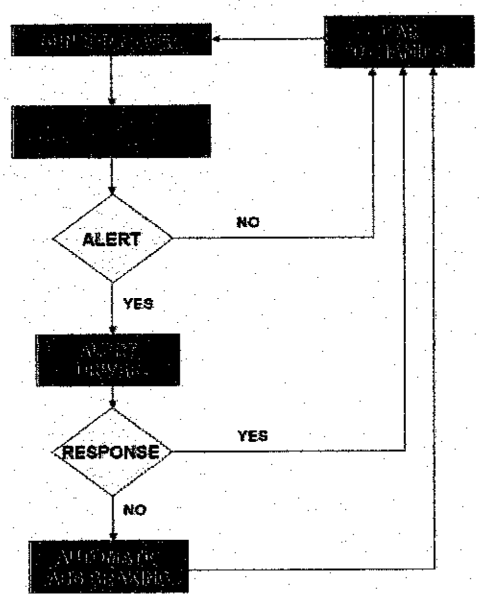


Figure 5 : Process flow of the system

Figure 5 shows the process flow of the Car's Object Detection and Avoidance Simulation using Verilog (CODA). It is very important that this algorithm is tested thoroughly before being used. All the possible conditions will be included in the test program. The Central Electronics Processor or previously described as IPU is the key towards the success of this project. The processor must be able to interprete the data accurately before making the split-microsecond decision. The robustness and the accuracy of the system come directly from the performance of the processor. All the logics must be properly defined and tested. Decision of engaging the ABS is very complex and needs a lot of verification process. It would not be easy to really determine and command a car that loses it's control and stability.

4.3 Develop behavioral code

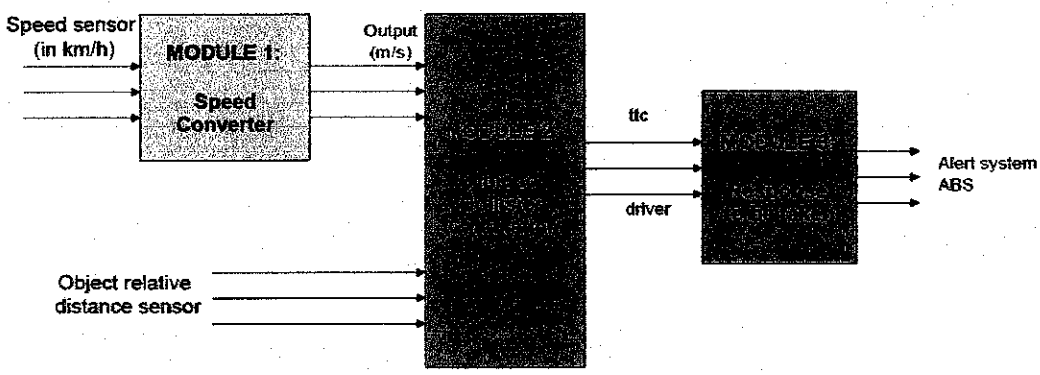


Figure 6 : Modules of the System

Figure 6 shows the entire module of the system which are determined base on the process flow. The modules consist of three module which will be explained in the details :

4.3.1 Speed Converter Module

The information of the vehicle's speed which is fed from the speed sensor is in km/h. This parameter will be converted to m/s in order to standardize the measurement parameters. The output of the module (in m/s) will then be fed to the next module; time to collision (ttc) module.

Generally, the speed converter module will start to convert the input speed measurement (in km/h) into m/s when reset = 1 and clock cycle is at positive edge *(please refer to the truth table in the next page)*. Speed in km/h will be converted to m/s using the formula :

$$spdms = spdkh * 10/36$$

4.3.1.1 Truth table for Speed Converter Module

Table 1 : Truth table of Speed Converter Module

Reset	**Clock**	**spdms**	**spdkh**
0	Positive edge	0	0
0	Negative edge	0	0
1	Positive edge	spdkh * 10/36	input
1	Negative edge	0	0

4.3.2 Time to Collision (ttc) Module

Combining the 'converted' speed measurement (form speed converter module) and the input from the object relative distance sensor, anticipated time of collision to happen is calculated based on the formula :

time to collision = distance / speed

For the Time to Collision analysis, the following condition is followed:

Maximum distance = 10m

Maximum speed = 200 km/h (55.56 m/s)

From the calculation of time to collision (ttc) using formula above, the ttc will be categorized into five different category, which will taken different type of reaction for the next module (Response to be taken Module). The categories are :

Table 2 : Category based on time to collision value

Category	**ttc (sec)**
A	0 – 0.10
B	0.10 – 0.20
C	0.20 – 0.30
D	0.30 – 0.40
E	0.40 – 0.90

Table 3 : Summary of time to collision with the respected category

Speed / Distance	9 – 10	8 – 9	7 – 8	6 – 7	5 – 6	4 – 5	3 – 4	2 – 3	1 – 2	0 - 1
36.11 – 55.56	0.18	0.16	0.14	0.13	0.11	0.09	0.07	0.05	0.04	0.02
25 – 36.11	0.28	0.25	0.22	0.19	0.17	0.14	0.11	0.08	0.06	0.03
16.67 – 25	0.40	0.36	0.32	0.28	0.24	0.20	0.16	0.12	0.08	0.04
11.11 – 16.67	0.60	0.54	0.48	0.42	0.36	0.30	0.24	0.18	0.12	0.06
0 – 11.11	0.90	0.81	0.72	0.63	0.54	0.45	0.36	0.27	0.18	0.09

Category A
Category B
Category C
Category D
Category E

4.3.3 Response to be taken

There are two types of action to be taken in this project which are the Antilock Braking System (ABS) and the driver alert system. These responses will be only initiated if there are no responses or action taken by the driver. If there are any action or reaction by the driver, the system will be disengaged until the next 'collision to be' moment.

The action of the ABS is categorized into five categories which are differs in the force to be applied onto the brake. The category is based on the time to collision category determined from the previous module (time to collision module).

Practically, a set of actuator system will be implemented on the braking system and the response to be taken determined by the system shall be translated in term of voltage values. The actuator response will be based on the output voltage values.

4.4 Simulation

The following are the simulation result of each module. The test values for each module are as shown in the table before each of the simulation figure:

4.4.1 Simulation result of Speed Converter Module

Table 4 : Test set values for the Speed Converter Module

spdkh	spdms
50	13.89
90	25.00
120	33.33

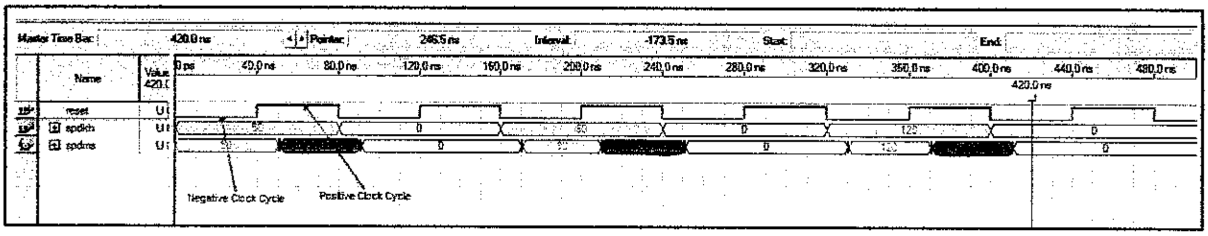


Figure 7 : Simulation of Speed Converter Module

The waveform in blue shows the value of the input for the speed converter module which is the speed in km/h (spdkh). On the other hand, the waveform in yellow and red colour indicates the output of the module which is the converted measurement of the speed from km/h into speed in m/s (spdms). Based on the module design, the input, will be converted into spdms only when the reset if at the positive edge, otherwise, the spdkh will be converted.

Based on Figure 9, when the reset is at the positive edge, the output waveform of spdms shows the expected value of the converted spdkh. In this project, only integer value is taken into consideration since it is easier to design and troubleshoot. During the negative cycle of the, the spdms value is the same as the spdkh value.

4.4.2 Simulation result of Time to Collision Module

Table 5 : Test set values for the Time to Collision Module

spdms	distance	ttc	category
10	5	0.50	A (1)
30	5	0.17	B (4)
50	5	0.10	C (5)

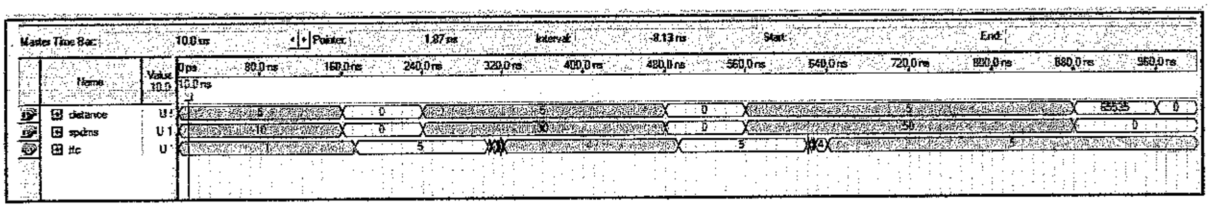


Figure 8 : Simulation of Time to Collision Module

The time to collision (ttc) is calculated based on the formula :

$$ttc = distance/speed$$

In this module, once the ttc is calculated, it will be categorized into five different categories based on ttc value. The smaller the value of ttc the higher the category will be. For instance, in Table 5, the value shows the decreases value of ttc but increases value of the category.

Based on the waveform in Figure 10, the ttc shows the correct category as stated in Table 5. There are some delay in the process of determine the category of the ttc. Referring to the orange waveform, there is lagging between each of the new set of input. This is because of the propagation delay occurred within the module.

4.4.3 Simulation result of Action Taken Module

Table 6 : Test set values for the Action Taken Module

driver	ttc	brake	alarm
0	1	5	1
1	1	0	0
0	2	4	1
1	2	0	0

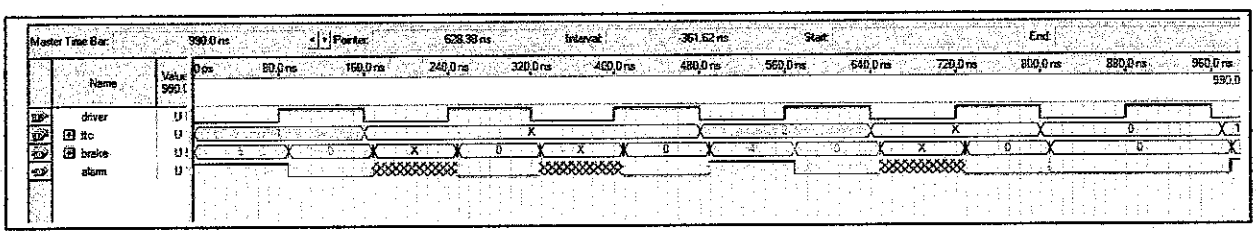


Figure 9 : Simulation of Action Taken Module

Action to be taken will be initiated if only there is no response form the driver. In this condition, the system will overtake the driver in order to avoid the car to be collided. Based on Table 6, it shown that whenever there is any action taken by the driver (which is indicate as in state "1"), both of the output will be at "0" state, which means the system did not overtake the driver role.

Referring to the waveform above, if the driver state is "0", the output; brake will be varies and the output alarm will be "1". The brake output varies depends on the category of time to collision (ttc) obtained from the previous module (time to collision module). The higher the category of ttc the smaller the brake value. The brake category determine of how much forces to be applied onto the brake (which will be carried out by a set of actuator).

4.4.4 Simulation result of the overall system : Car's Object Detection and Avoidance (CODA)

Table 7 : Test set values for CODA

reset	spdkh	distance	driver	brake	alarm
1	50	5	0	1	1
0	50	5	0	4	1
1	50	5	1	0	0
0	50	5	1	0	0

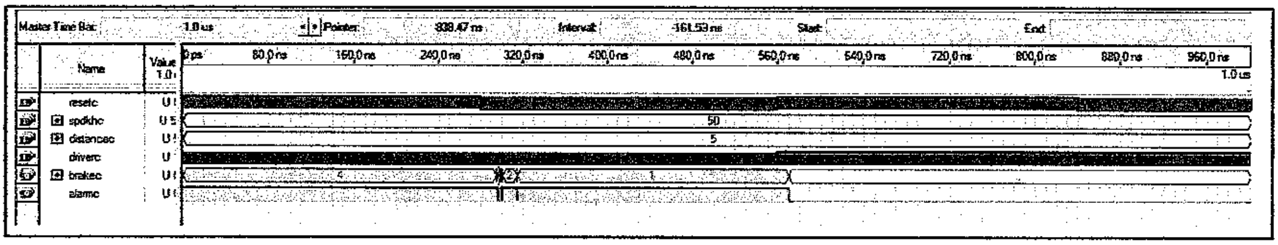


Figure 10 : Simulation of CODA

Figure 12 shows the simulation result of the completed module of the Car's Object Detection and Avoidance system. With speed in km/h (spdkh) and the distance to be constant, driver action will determine the system response.

As shown in above figure, the waveform in blue colour shows that the driving control is overtake by the system whilst the waveform in yellow colour shows no response by the system. This is because of the driver action at particular condition. It shows that if the driver is not taking any action, then only the system will play it's role.

The reset value will only determined whether the spdkh will be converted into spdms. This operation occur within the speed converter module, where in this 'big picture' of the overall module, the operation is not shown.

CHAPTER 5
CONCLUSION AND RECOMMENDATION

5.1 Conclusion

The objective of the project which is to simulate the Car's Object Detection and Avoidance is achieved. The designated system however is a very basic system which does not consider the actual and real life situation. The idea is not to develop a practical system but more to get familiar with the process and procedure on how to design a system which can be implemented on an intelligent circuit (IC). Further improvement is needed in order to get this project to be practical and marketable.

5.2 Recommendation

It is recommended that the data and information used in the design process to be based on the reliable and accurate. In this project, all the data and information used in the design process is generally based on assumption which is not good if the system is design for marketable purpose.

REFERENCES

[1] Wikipedia, The Free Encyclopedia, "Accident Statistic" (26/8/2006)
http://en.wikipedia.org/wiki/Road_accident_statistics_on_a_model-by-model_basis

[2] Wikipedia, The Free Encyclopedia, "Car Accident" (26/8/2006)
http://en.wikipedia.org/wiki/Car_accident

[3] eSafety Support, European Commission funded project coordinated by ERTICO, (13/1/07)
http://www.esafetysupport.org/en/learn_about_esafety_systems/esafety_systems/esafety_systems.htm

[4] United States Department of Transportation – Intelligent Transport System, Cooperative Intersection Collision Avoidance System, (13/1/07)
http://www.its.dot.gov/cicas/index.htm

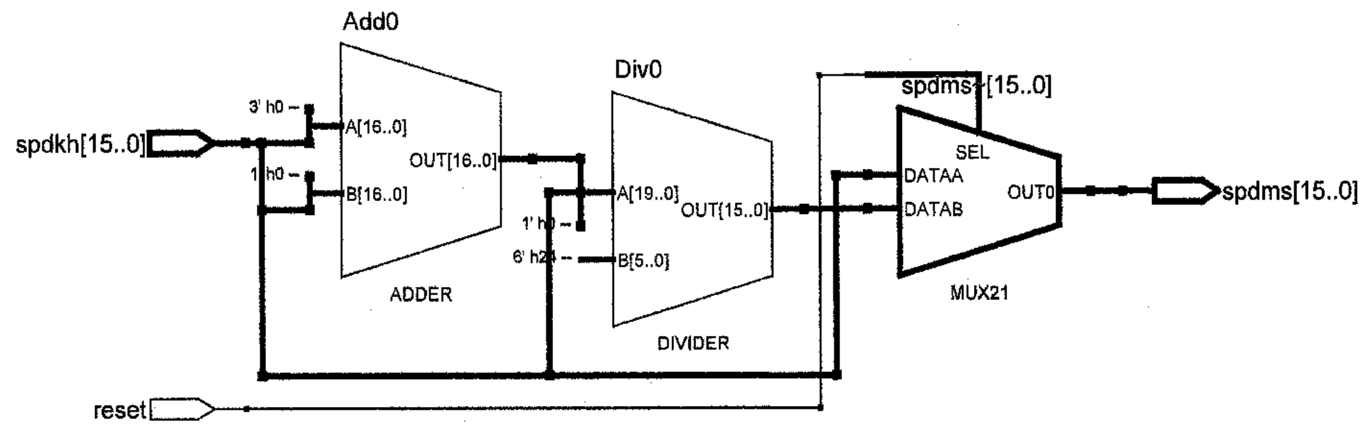
[5] Verilog Introduction for Digital Design, Dr Hyde of Bucknell University, Lewisburg PA, (18/9/07)
http://www.see.ed.ac.uk/~gerard/Teach/Verilog/index.html?http://oldeee.see.ed.ac.uk/~gerard/Teach/Verilog/index.html

[6] Emerald System, IC Design Services/EDA & PCB Design Tools, Digital Design Training, Universiti Teknologi PETRONAS Training Material.

APPENDICES

APPENDIX A

```
le speedconverter (reset,spdkh,spdms);

t reset;
t [15:0]spdkh;
ut [15:0]spdms;
[15:0]spdms;


meter a = 10;
meter b = 36;
:d is converted only if clock and reset = 1

ys @(spdkh or reset)
if (reset == 1)
    spdms = (spdkh * a)/b;
else
    spdms = spdkh;

.odule
```


Add0
Div0
spdms~[15..0]
3' h0
1' h0
spdkh[15..0]
A[16..0]
B[16..0]
OUT[16..0]
A[19..0]
B[5..0]
OUT[15..0]
1' h0
6' h24
SEL
DATAA
DATAB
OUT0
spdms[15..0]
ADDER
DIVIDER
MUX21
reset

Resource	Usage
Total logic elements	264
-- Combinational with no register	264
-- Register only	0
-- Combinational with a register	0
Logic element usage by number of LUT inputs	
-- 4 input functions	39
-- 3 input functions	84
-- 2 input functions	83
-- 1 input functions	58
-- 0 input functions	0
-- Combinational cells for routing	0
Logic elements by mode	
-- normal mode	160
-- arithmetic mode	104
-- qfbk mode	0
-- register cascade mode	0
-- synchronous clear/load mode	0
-- asynchronous clear/load mode	0
Total registers	0
Total logic cells in carry chains	134
I/O pins	33
Maximum fan-out node	reset
Maximum fan-out	16
Total fan-out	648
Average fan-out	2.18

Option	Setting	Default Value
Use smart compilation	Off	Off
Generate Serial Vector Format File (.svf) for Target Device	Off	Off
Generate a JEDEC STAPL Format File (.jam) for Target Device	Off	Off
Generate an uncompressed Jam STAPL Byte Code 2.0 File (.jbc) for Target Device	Off	Off
Generate a compressed Jam STAPL Byte Code 2.0 File (.jbc) for Target Device	On	On
Compression mode	Off	Off
Clock source for configuration device	Internal	Internal
Clock frequency of the configuration device	10 MHZ	10 MHz
Divide clock frequency by	1	1
JTAG user code for target device	Ffffffff	Ffffffff
Auto user code	Off	Off
Use configuration device	On	On
Configuration device	Auto	Auto
JTAG user code for configuration device	Ffffffff	Ffffffff
Configuration device auto user code	Off	Off
Auto-increment JTAG user code for multiple configuration devices	On	On
Disable CONF_DONE and nSTATUS pull-ups on configuration device	Off	Off
Generate Tabular Text File (.ttf) For Target Device	Off	Off
Generate Raw Binary File (.rbf) For Target Device	Off	Off
Generate Hexadecimal (Intel-Format) Output File (.hexout) for Target Device	Off	Off
Hexadecimal Output File start address	0	0
Hexadecimal Output File count direction	Up	Up
Release clears before tri-states	Off	Off
Auto-restart configuration after error	On	On

Option	Setting	Default Value
Device	AUTO	
Use smart compilation	Off	Off
Router Timing Optimization Level	Normal	Normal
Placement Effort Multiplier	1.0	1.0
Router Effort Multiplier	1.0	1.0
Optimize Hold Timing	IO Paths and Minimum TPD Paths	IO Paths and Minimum TPD Paths
Optimize Fast-Corner Timing	Off	Off
Optimize Timing	Normal compilation	Normal compilation
Optimize IOC Register Placement for Timing	On	On
Limit to One Fitting Attempt	Off	Off
Final Placement Optimizations	Automatically	Automatically
Fitter Aggressive Routability Optimizations	Automatically	Automatically
Fitter Initial Placement Seed	1	1
Slow Slew Rate	Off	Off
PCI I/O	Off	Off
Weak Pull-Up Resistor	Off	Off
Enable Bus-Hold Circuitry	Off	Off
Auto Global Memory Control Signals	Off	Off
Auto Packed Registers -- Stratix/Stratix GX	Auto	Auto
Auto Delay Chains	On	On
Auto Merge PLLs	On	On
Perform Physical Synthesis for Combinational Logic	Off	Off
Perform Register Duplication	Off	Off
Perform Register Retiming	Off	Off
Perform Asynchronous Signal Pipelining	Off	Off
Fitter Effort	Auto Fit	Auto Fit
Physical Synthesis Effort Level	Normal	Normal
Logic Cell Insertion - Logic Duplication	Auto	Auto
Auto Register Duplication	Auto	Auto
Auto Global Clock	On	On
Auto Global Register Control Signals	On	On

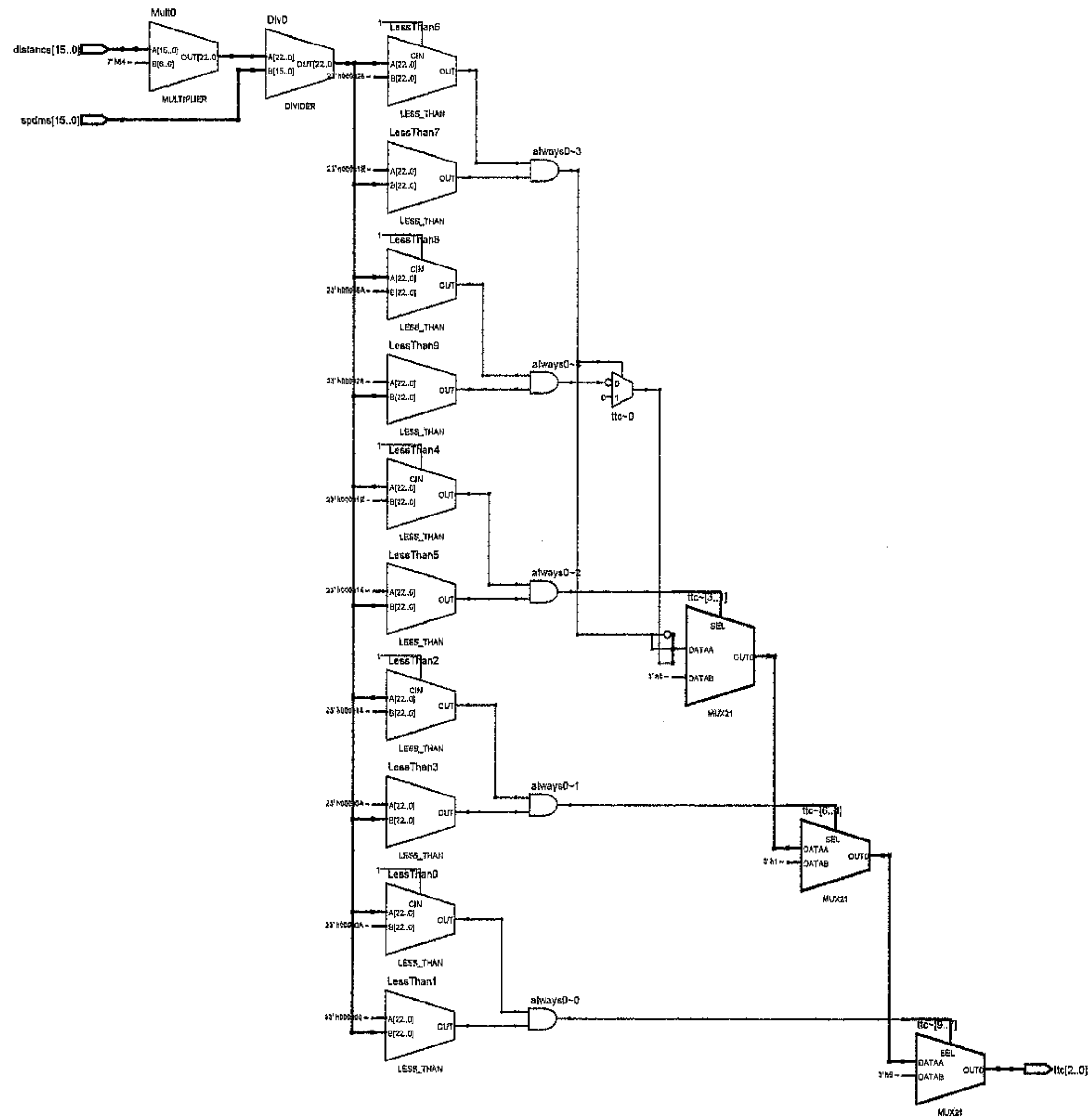
Option	Setting	From	To	Entity Name
Device Name	EP1S10F484C5			
Timing Models	Final			
Number of source nodes to report per destination node	10			
Number of destination nodes to report	10			
Number of paths to report	200			
Report Minimum Timing Checks	Off			
Use Fast Timing Models	Off			
Report IO Paths Separately	Off			
Default hold multicycle	Same As Multicycle			
Cut paths between unrelated clock domains	On			
Cut off read during write signal paths	On			
Cut off feedback from I/O pins	On			
Report Combined Fast/Slow Timing	Off			
Ignore Clock Settings	Off			
Analyze latches as synchronous elements	On			
Enable Recovery/Removal analysis	Off			
Enable Clock Latency	Off			
Use TimeQuest Timing Analyzer	Off			

Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock
Worst-case tpd	N/A	None	64.875 ns	spdkh[2]	spdms[0]	--	--
Total number of failed paths							

Failed Paths
0
0

APPENDIX B

```
le timetocollision (spdms, distance, ttc);

t [15:0]spdms;
t [15:0]distance;

ıt [2:0] ttc;
[2:0] ttc;
15:0] w;
neter h = 100;

distance*h)/spdms;            //time to collision formula


s divided into 5 category base on the designated time interval
ys @ (spdms or distance)
begin

if
    ((distance*h)/spdms <= 10 & (distance*h)/spdms > 0) begin
    ttc = 5; end                        // category A
else if
    ((distance*h)/spdms <= 20 & (distance*h)/spdms > 10) begin
    ttc = 4; end                        // Category B
else if
    ((distance*h)/spdms <= 30 & (distance*h)/spdms > 20) begin
    ttc = 3; end                        // Category C
else if
    ((distance*h)/spdms <= 40 & (distance*h)/spdms > 30) begin
    ttc = 2; end                        // Category D
else if
    ((distance*h)/spdms <= 90 & (distance*h)/spdms > 40) begin
    ttc = 1; end                        // Category E
else
    ttc = 5;
end

odule
```


Mult0
distance[15..0]
spdms[15..0]
MULTIPLIER
Div0
DIVIDER
LessThan6
LessThan7
LessThan8
LessThan9
LessThan4
LessThan5
LessThan2
LessThan3
LessThan0
LessThan1
LESS_THAN
always0~3
always0~2
always0~1
always0~0
ttc~0
MUX21
ttc[2..0]

Resource	Usage
Total logic elements	621
-- Combinational with no register	621
-- Register only	0
-- Combinational with a register	0
Logic element usage by number of LUT inputs	
-- 4 input functions	105
-- 3 input functions	434
-- 2 input functions	57
-- 1 input functions	25
-- 0 input functions	0
-- Combinational cells for routing	0
Logic elements by mode	
-- normal mode	340
-- arithmetic mode	281
-- qfbk mode	0
-- register cascade mode	0
-- synchronous clear/load mode	0
-- asynchronous clear/load mode	0
Total registers	0
Total logic cells in carry chains	306
I/O pins	35
Maximum fan-out node	spdms[15]
Maximum fan-out	27
Total fan-out	1864
Average fan-out	2.84

Option	Setting	Default Value
Use smart compilation	Off	Off
Generate Serial Vector Format File (.svf) for Target Device	Off	Off
Generate a JEDEC STAPL Format File (.jam) for Target Device	Off	Off
Generate an uncompressed Jam STAPL Byte Code 2.0 File (.jbc) for Target Device	Off	Off
Generate a compressed Jam STAPL Byte Code 2.0 File (.jbc) for Target Device	On	On
Compression mode	Off	Off
Clock source for configuration device	Internal	Internal
Clock frequency of the configuration device	10 MHZ	10 MHz
Divide clock frequency by	1	1
JTAG user code for target device	Fffffff	Fffffff
Auto user code	Off	Off
Use configuration device	On	On
Configuration device	Auto	Auto
JTAG user code for configuration device	Fffffff	Fffffff
Configuration device auto user code	Off	Off
Auto-increment JTAG user code for multiple configuration devices	On	On
Disable CONF_DONE and nSTATUS pull-ups on configuration device	Off	Off
Generate Tabular Text File (.ttf) For Target Device	Off	Off
Generate Raw Binary File (.rbf) For Target Device	Off	Off
Generate Hexadecimal (Intel-Format) Output File (.hexout) for Target Device	Off	Off
Hexadecimal Output File start address	0	0
Hexadecimal Output File count direction	Up	Up
Release clears before tri-states	Off	Off
Auto-restart configuration after error	On	On

Option	Setting	Default Value
Device	AUTO	
Use smart compilation	Off	Off
Router Timing Optimization Level	Normal	Normal
Placement Effort Multiplier	1.0	1.0
Router Effort Multiplier	1.0	1.0
Optimize Hold Timing	IO Paths and Minimum TPD Paths	IO Paths and Minimum TPD Paths
Optimize Fast-Corner Timing	Off	Off
Optimize Timing	Normal compilation	Normal compilation
Optimize IOC Register Placement for Timing	On	On
Limit to One Fitting Attempt	Off	Off
Final Placement Optimizations	Automatically	Automatically
Fitter Aggressive Routability Optimizations	Automatically	Automatically
Fitter Initial Placement Seed	1	1
Slow Slew Rate	Off	Off
PCI I/O	Off	Off
Weak Pull-Up Resistor	Off	Off
Enable Bus-Hold Circuitry	Off	Off
Auto Global Memory Control Signals	Off	Off
Auto Packed Registers -- Stratix/Stratix GX	Auto	Auto
Auto Delay Chains	On	On
Auto Merge PLLs	On	On
Perform Physical Synthesis for Combinational Logic	Off	Off
Perform Register Duplication	Off	Off
Perform Register Retiming	Off	Off
Perform Asynchronous Signal Pipelining	Off	Off
Fitter Effort	Auto Fit	Auto Fit
Physical Synthesis Effort Level	Normal	Normal
Logic Cell Insertion - Logic Duplication	Auto	Auto
Auto Register Duplication	Auto	Auto
Auto Global Clock	On	On
Auto Global Register Control Signals	On	On

Option	Setting	From	To	Entity Name
Device Name	EP1S10F484C5			
Timing Models	Final			
Number of source nodes to report per destination node	10			
Number of destination nodes to report	10			
Number of paths to report	200			
Report Minimum Timing Checks	Off			
Use Fast Timing Models	Off			
Report IO Paths Separately	Off			
Default hold multicycle	Same As Multicycle			
Cut paths between unrelated clock domains	On			
Cut off read during write signal paths	On			
Cut off feedback from I/O pins	On			
Report Combined Fast/Slow Timing	Off			
Ignore Clock Settings	Off			
Analyze latches as synchronous elements	On			
Enable Recovery/Removal analysis	Off			
Enable Clock Latency	Off			
Use TimeQuest Timing Analyzer	Off			

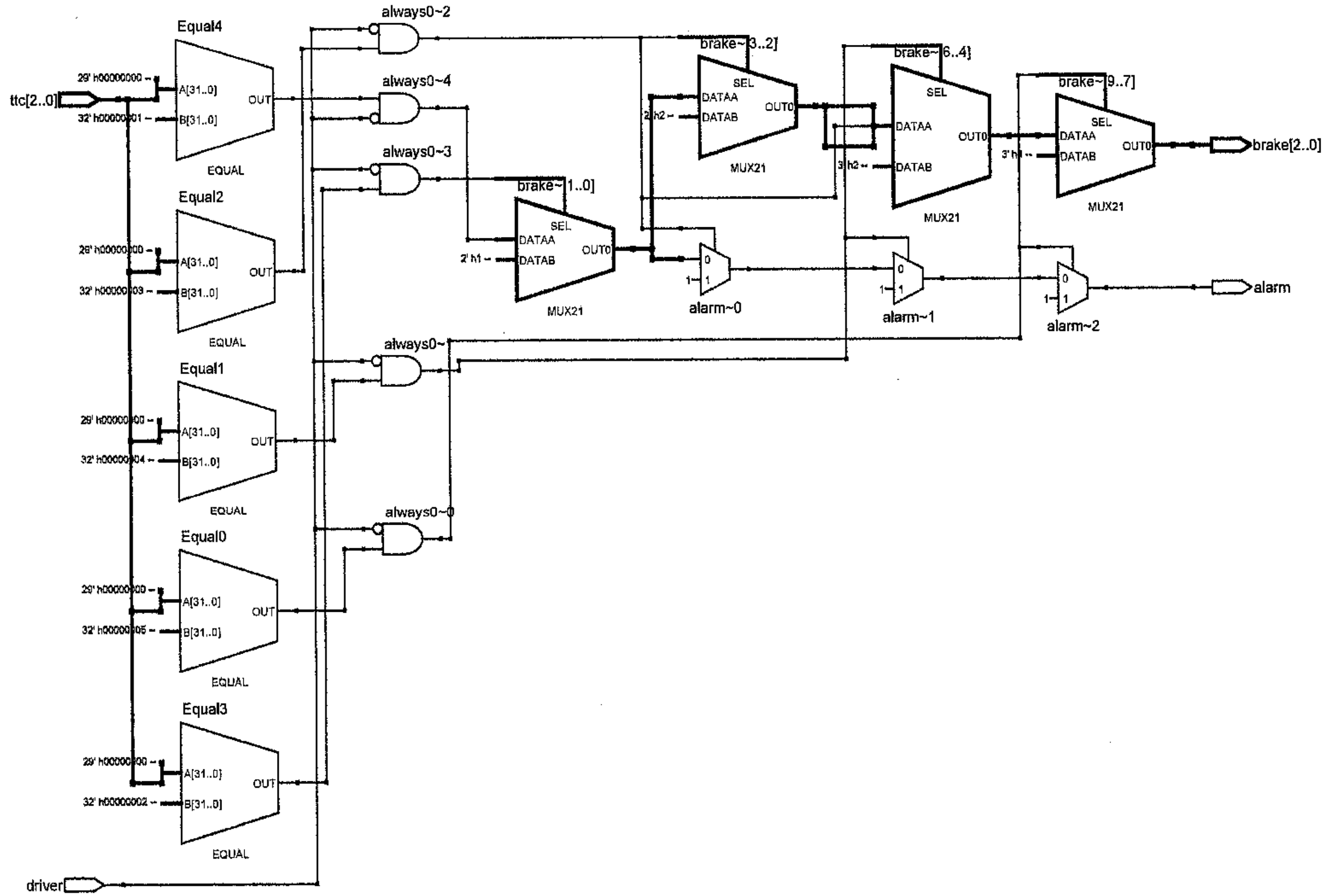
Slack	Required P2P Time	Actual P2P Time	From	To
N/A	None	110.964 ns	spdms[12]	ttc[1]
N/A	None	110.702 ns	spdms[12]	ttc[0]
N/A	None	110.571 ns	spdms[12]	ttc[2]
N/A	None	110.542 ns	spdms[13]	ttc[1]
N/A	None	110.423 ns	spdms[14]	ttc[1]
N/A	None	110.280 ns	spdms[13]	ttc[0]
N/A	None	110.161 ns	spdms[14]	ttc[0]
N/A	None	110.149 ns	spdms[13]	ttc[2]
N/A	None	110.030 ns	spdms[14]	ttc[2]
N/A	None	109.643 ns	spdms[15]	ttc[1]
N/A	None	109.381 ns	spdms[15]	ttc[0]
N/A	None	109.250 ns	spdms[15]	ttc[2]
N/A	None	109.185 ns	distance[0]	ttc[1]
N/A	None	109.035 ns	distance[3]	ttc[1]
N/A	None	108.923 ns	distance[0]	ttc[0]
N/A	None	108.919 ns	spdms[10]	ttc[1]
N/A	None	108.797 ns	distance[2]	ttc[1]
N/A	None	108.792 ns	distance[0]	ttc[2]
N/A	None	108.773 ns	distance[3]	ttc[0]
N/A	None	108.657 ns	spdms[10]	ttc[0]
N/A	None	108.642 ns	distance[3]	ttc[2]
N/A	None	108.595 ns	distance[1]	ttc[1]
N/A	None	108.535 ns	distance[2]	ttc[0]
N/A	None	108.526 ns	spdms[10]	ttc[2]
N/A	None	108.417 ns	spdms[11]	ttc[1]
N/A	None	108.404 ns	distance[2]	ttc[2]
N/A	None	108.333 ns	distance[1]	ttc[0]
N/A	None	108.202 ns	distance[1]	ttc[2]
N/A	None	108.155 ns	spdms[11]	ttc[0]
N/A	None	108.040 ns	distance[7]	ttc[1]
N/A	None	108.024 ns	spdms[11]	ttc[2]
N/A	None	107.994 ns	spdms[9]	ttc[1]
N/A	None	107.778 ns	distance[7]	ttc[0]
N/A	None	107.732 ns	spdms[9]	ttc[0]
N/A	None	107.691 ns	distance[4]	ttc[1]
N/A	None	107.647 ns	distance[7]	ttc[2]
N/A	None	107.601 ns	spdms[9]	ttc[2]
N/A	None	107.535 ns	distance[9]	ttc[1]
N/A	None	107.507 ns	distance[8]	ttc[1]
N/A	None	107.432 ns	distance[6]	ttc[1]
N/A	None	107.429 ns	distance[4]	ttc[0]
N/A	None	107.407 ns	distance[10]	ttc[1]
N/A	None	107.387 ns	distance[5]	ttc[1]
N/A	None	107.298 ns	distance[4]	ttc[2]
N/A	None	107.273 ns	distance[9]	ttc[0]
N/A	None	107.267 ns	distance[11]	ttc[1]
N/A	None	107.245 ns	distance[8]	ttc[0]
N/A	None	107.230 ns	distance[13]	ttc[1]

N/A	None	107.170 ns	distance[6] ttc[0]
N/A	None	107.145 ns	distance[10]ttc[0]
N/A	None	107.142 ns	distance[9] ttc[2]
N/A	None	107.125 ns	distance[5] ttc[0]
N/A	None	107.114 ns	distance[8] ttc[2]
N/A	None	107.113 ns	distance[12]ttc[1]
N/A	None	107.085 ns	spdms[8] ttc[1]
N/A	None	107.039 ns	distance[6] ttc[2]
N/A	None	107.014 ns	distance[10]ttc[2]
N/A	None	107.005 ns	distance[11]ttc[0]
N/A	None	106.994 ns	distance[5] ttc[2]
N/A	None	106.968 ns	distance[13]ttc[0]
N/A	None	106.902 ns	distance[14]ttc[1]
N/A	None	106.874 ns	distance[11]ttc[2]
N/A	None	106.874 ns	distance[15]ttc[1]
N/A	None	106.851 ns	distance[12]ttc[0]
N/A	None	106.837 ns	distance[13]ttc[2]
N/A	None	106.823 ns	spdms[8] ttc[0]
N/A	None	106.811 ns	spdms[7] ttc[1]
N/A	None	106.720 ns	distance[12]ttc[2]
N/A	None	106.692 ns	spdms[8] ttc[2]
N/A	None	106.640 ns	distance[14]ttc[0]
N/A	None	106.612 ns	distance[15]ttc[0]
N/A	None	106.549 ns	spdms[7] ttc[0]
N/A	None	106.509 ns	distance[14]ttc[2]
N/A	None	106.481 ns	distance[15]ttc[2]
N/A	None	106.418 ns	spdms[7] ttc[2]
N/A	None	105.508 ns	spdms[6] ttc[1]
N/A	None	105.374 ns	spdms[4] ttc[1]
N/A	None	105.344 ns	spdms[5] ttc[1]
N/A	None	105.246 ns	spdms[6] ttc[0]
N/A	None	105.115 ns	spdms[6] ttc[2]
N/A	None	105.112 ns	spdms[4] ttc[0]
N/A	None	105.082 ns	spdms[5] ttc[0]
N/A	None	104.981 ns	spdms[4] ttc[2]
N/A	None	104.951 ns	spdms[5] ttc[2]
N/A	None	103.756 ns	spdms[3] ttc[1]
N/A	None	103.508 ns	spdms[2] ttc[1]
N/A	None	103.494 ns	spdms[3] ttc[0]
N/A	None	103.363 ns	spdms[3] ttc[2]
N/A	None	103.246 ns	spdms[2] ttc[0]
N/A	None	103.204 ns	spdms[1] ttc[1]
N/A	None	103.115 ns	spdms[2] ttc[2]
N/A	None	102.942 ns	spdms[1] ttc[0]
N/A	None	102.811 ns	spdms[1] ttc[2]
N/A	None	102.520 ns	spdms[0] ttc[1]
N/A	None	102.258 ns	spdms[0] ttc[0]
N/A	None	102.127 ns	spdms[0] ttc[2]

APPENDIX C

```
le actiontaken (ttc, driver, brake, alarm);

t [2:0] ttc;
t driver;


ıt [2:0]brake;
ıt alarm;
[2:0] brake;
ılarm;


ys @ (ttc or driver)

ı


if  (ttc == 5 & driver == 0) begin          // Category A
    brake = 1;                               // 80% force applied on the brake
    alarm = 1; end                           // alarm activated

else if

    (ttc == 4 & driver == 0) begin           // Category B
    brake = 2;                               // 60% force applied on the brake
    alarm = 1; end                           // alarm activated

else if

    (ttc == 3 & driver == 0) begin           // Category C
    brake = 3;                               // 40% force applied on the brake
    alarm = 1; end                           // alarm activated

else if

    (ttc == 2 & driver == 0) begin           // Category D
    brake = 4;                               // 30% force applied on the brake
    alarm = 1; end                           // alarm activated

else if

    (ttc == 1 & driver == 0) begin           // Category E
    brake = 5;                               // 10% force applied on the brake
    alarm = 1; end                           // alarm activated

else

    begin
        brake = 0;
        alarm = 0;
    end
```

```
end

odule
```


ttc[2..0]
driver
Equal4
Equal2
Equal1
Equal0
Equal3
A[31..0]
B[31..0]
OUT
EQUAL
always0~2
always0~4
always0~3
always0~
always0~0
brake~[1..0]
brake~[3..2]
brake~[6..4]
brake~[9..7]
SEL
DATAA
DATAB
OUT0
MUX21
2' h1
2' h2
3' h2
alarm~0
alarm~1
alarm~2
brake[2..0]
alarm

Resource	Usage
Total logic elements	4
-- Combinational with no register	4
-- Register only	0
-- Combinational with a register	0
Logic element usage by number of LUT inputs	
-- 4 input functions	4
-- 3 input functions	0
-- 2 input functions	0
-- 1 input functions	0
-- 0 input functions	0
-- Combinational cells for routing	0
Logic elements by mode	
-- normal mode	4
-- arithmetic mode	0
-- qfbk mode	0
-- register cascade mode	0
-- synchronous clear/load mode	0
-- asynchronous clear/load mode	0
Total registers	0
I/O pins	8
Maximum fan-out node	ttc[0]
Maximum fan-out	4
Total fan-out	20
Average fan-out	1.67

Option	Setting	Default Value
Use smart compilation	Off	Off
Generate Serial Vector Format File (.svf) for Target Device	Off	Off
Generate a JEDEC STAPL Format File (.jam) for Target Device	Off	Off
Generate an uncompressed Jam STAPL Byte Code 2.0 File (.jbc) for Target Device	Off	Off
Generate a compressed Jam STAPL Byte Code 2.0 File (.jbc) for Target Device	On	On
Compression mode	Off	Off
Clock source for configuration device	Internal	Internal
Clock frequency of the configuration device	10 MHZ	10 MHz
Divide clock frequency by	1	1
JTAG user code for target device	Ffffffff	Ffffffff
Auto user code	Off	Off
Use configuration device	On	On
Configuration device	Auto	Auto
JTAG user code for configuration device	Ffffffff	Ffffffff
Configuration device auto user code	Off	Off
Auto-increment JTAG user code for multiple configuration devices	On	On
Disable CONF_DONE and nSTATUS pull-ups on configuration device	Off	Off
Generate Tabular Text File (.ttf) For Target Device	Off	Off
Generate Raw Binary File (.rbf) For Target Device	Off	Off
Generate Hexadecimal (Intel-Format) Output File (.hexout) for Target Device	Off	Off
Hexadecimal Output File start address	0	0
Hexadecimal Output File count direction	Up	Up
Release clears before tri-states	Off	Off
Auto-restart configuration after error	On	On

Option	Setting	Default Value
Device	AUTO	
Use smart compilation	Off	Off
Router Timing Optimization Level	Normal	Normal
Placement Effort Multiplier	1.0	1.0
Router Effort Multiplier	1.0	1.0
Optimize Hold Timing	IO Paths and Minimum TPD Paths	IO Paths and Minimum TPD Paths
Optimize Fast-Corner Timing	Off	Off
Optimize Timing	Normal compilation	Normal compilation
Optimize IOC Register Placement for Timing	On	On
Limit to One Fitting Attempt	Off	Off
Final Placement Optimizations	Automatically	Automatically
Fitter Aggressive Routability Optimizations	Automatically	Automatically
Fitter Initial Placement Seed	1	1
Slow Slew Rate	Off	Off
PCI I/O	Off	Off
Weak Pull-Up Resistor	Off	Off
Enable Bus-Hold Circuitry	Off	Off
Auto Global Memory Control Signals	Off	Off
Auto Packed Registers -- Stratix/Stratix GX	Auto	Auto
Auto Delay Chains	On	On
Auto Merge PLLs	On	On
Perform Physical Synthesis for Combinational Logic	Off	Off
Perform Register Duplication	Off	Off
Perform Register Retiming	Off	Off
Perform Asynchronous Signal Pipelining	Off	Off
Fitter Effort	Auto Fit	Auto Fit
Physical Synthesis Effort Level	Normal	Normal
Logic Cell Insertion - Logic Duplication	Auto	Auto
Auto Register Duplication	Auto	Auto
Auto Global Clock	On	On
Auto Global Register Control Signals	On	On

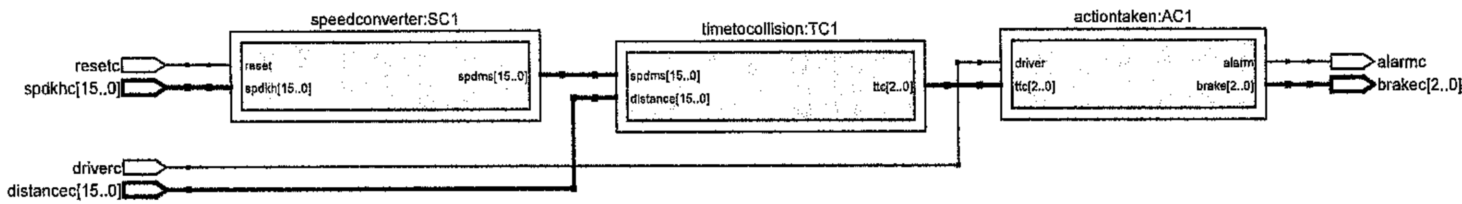
Option	Setting	From	To	Entity Name
Device Name	EP1S10F484C5			
Timing Models	Final			
Number of source nodes to report per destination node	10			
Number of destination nodes to report	10			
Number of paths to report	200			
Report Minimum Timing Checks	Off			
Use Fast Timing Models	Off			
Report IO Paths Separately	Off			
Default hold multicycle	Same As Multicycle			
Cut paths between unrelated clock domains	On			
Cut off read during write signal paths	On			
Cut off feedback from I/O pins	On			
Report Combined Fast/Slow Timing	Off			
Ignore Clock Settings	Off			
Analyze latches as synchronous elements	On			
Enable Recovery/Removal analysis	Off			
Enable Clock Latency	Off			
Use TimeQuest Timing Analyzer	Off			

Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
Worst-case tpd	N/A	None	8.751 ns	ttc[2]	alarm	--	--	0
Total number of failed paths								0

Slack	Required P2P Time	Actual P2P Time	From To
N/A	None	8.751 ns	ttc[2] alarm
N/A	None	8.748 ns	ttc[2] brake[2]
N/A	None	8.745 ns	ttc[2] brake[0]
N/A	None	8.507 ns	ttc[1] brake[0]
N/A	None	8.501 ns	ttc[1] alarm
N/A	None	8.499 ns	ttc[1] brake[2]
N/A	None	8.426 ns	driverbrake[0]
N/A	None	8.420 ns	driveralarm
N/A	None	8.417 ns	driverbrake[2]
N/A	None	8.350 ns	ttc[2] brake[1]
N/A	None	8.240 ns	ttc[0] alarm
N/A	None	8.238 ns	ttc[0] brake[2]
N/A	None	8.235 ns	ttc[0] brake[0]
N/A	None	8.109 ns	ttc[1] brake[1]
N/A	None	8.024 ns	driverbrake[1]
N/A	None	7.840 ns	ttc[0] brake[1]

APPENDIX D

```
e coda (spdkhc, resetc, distancec, driverc, brakec, alarmc);

 [15:0] spdkhc;
 [15:0] distancec;
 resetc;
 driverc;

t [2:0] brakec;
t alarmc;

[15:0] w1;
[2:0] w2;

converter SC1 (resetc, spdkhc, w1);

ocollision TC1 (w1, distancec, w2);

ntaken AC1 (w2, driverc, brakec, alarmc);


dule
```


speedconverter:SC1
timetocollision:TC1
actiontaken:AC1
resetc
spdkhc[15..0]
driverc
distancec[15..0]
reset
spdkh[15..0]
spdms[15..0]
spdms[15..0]
distance[15..0]
ttc[2..0]
driver
ttc[2..0]
alarm
brake[2..0]
alarmc
brakec[2..0]

Option	Setting	Default Value
Top-level entity name	coda	coda
Family name	Stratix	Stratix
Use smart compilation	Off	Off
Restructure Multiplexers	Auto	Auto
Create Debugging Nodes for IP Cores	Off	Off
Preserve fewer node names	On	On
Disable OpenCore Plus hardware evaluation	Off	Off
Verilog Version	Verilog_2001	Verilog_2001
VHDL Version	VHDL93	VHDL93
State Machine Processing	Auto	Auto
Extract Verilog State Machines	On	On
Extract VHDL State Machines	On	On
Add Pass-Through Logic to Inferred RAMs	On	On
DSP Block Balancing	Auto	Auto
Maximum DSP Block Usage	Unlimited	Unlimited
NOT Gate Push-Back	On	On
Power-Up Don't Care	On	On
Remove Redundant Logic Cells	Off	Off
Remove Duplicate Registers	On	On
Ignore CARRY Buffers	Off	Off
Ignore CASCADE Buffers	Off	Off
Ignore GLOBAL Buffers	Off	Off
Ignore ROW GLOBAL Buffers	Off	Off
Ignore LCELL Buffers	Off	Off
Ignore SOFT Buffers	On	On
Limit AHDL Integers to 32 Bits	Off	Off
Optimization Technique -- Stratix/Stratix GX	Balanced	Balanced
Carry Chain Length -- Stratix/Stratix GX/Cyclone/MAX II/Cyclone II	70	70
Auto Carry Chains	On	On
Auto Open-Drain Pins	On	On
Remove Duplicate Logic	On	On
Perform WYSIWYG Primitive Resynthesis	Off	Off
Perform gate-level register retiming	Off	Off
Allow register retiming to trade off Tsu/Tco with Fmax	On	On
Auto ROM Replacement	On	On
Auto RAM Replacement	On	On
Auto DSP Block Replacement	On	On
Auto Shift Register Replacement	On	On
Auto Clock Enable Replacement	On	On
Allow Synchronous Control Signals	On	On
Force Use of Synchronous Clear Signals	Off	Off
Auto RAM Block Balancing	On	On
Auto Resource Sharing	Off	Off
Allow Any RAM Size For Recognition	Off	Off
Allow Any ROM Size For Recognition	Off	Off
Allow Any Shift Register Size For Recognition	Off	Off
Maximum Number of M512 Memory Blocks	Unlimited	Unlimited
Maximum Number of M4K Memory Blocks	Unlimited	Unlimited

Maximum Number of M-RAM Memory Blocks	Unlimited	Unlimited
Ignore translate_off and translate_on Synthesis Directives	Off	Off
Show Parameter Settings Tables in Synthesis Report	On	On
Ignore Maximum Fan-Out Assignments	Off	Off
Retiming Meta-Stability Register Sequence Length	2	2
PowerPlay Power Optimization	Normal compilation	Normal compilation
HDL message level	Level2	Level2

Option	Setting	Default Value
Device	AUTO	
Use smart compilation	Off	Off
Router Timing Optimization Level	Normal	Normal
Placement Effort Multiplier	1.0	1.0
Router Effort Multiplier	1.0	1.0
Optimize Hold Timing	IO Paths and Minimum TPD Paths	IO Paths and Minimum TPD Paths
Optimize Fast-Corner Timing	Off	Off
Optimize Timing	Normal compilation	Normal compilation
Optimize IOC Register Placement for Timing	On	On
Limit to One Fitting Attempt	Off	Off
Final Placement Optimizations	Automatically	Automatically
Fitter Aggressive Routability Optimizations	Automatically	Automatically
Fitter Initial Placement Seed	1	1
Slow Slew Rate	Off	Off
PCI I/O	Off	Off
Weak Pull-Up Resistor	Off	Off
Enable Bus-Hold Circuitry	Off	Off
Auto Global Memory Control Signals	Off	Off
Auto Packed Registers – Stratix/Stratix GX	Auto	Auto
Auto Delay Chains	On	On
Auto Merge PLLs	On	On
Perform Physical Synthesis for Combinational Logic	Off	Off
Perform Register Duplication	Off	Off
Perform Register Retiming	Off	Off
Perform Asynchronous Signal Pipelining	Off	Off
Fitter Effort	Auto Fit	Auto Fit
Physical Synthesis Effort Level	Normal	Normal
Logic Cell Insertion - Logic Duplication	Auto	Auto
Auto Register Duplication	Auto	Auto
Auto Global Clock	On	On
Auto Global Register Control Signals	On	On

Statistic	Value
Total registers	0
Number of registers using Synchronous Clear	0
Number of registers using Synchronous Load	0
Number of registers using Asynchronous Clear	0
Number of registers using Asynchronous Load	0
Number of registers using Clock Enable	0
Number of registers using Preset	0

Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock
Worst-case tpd	N/A	None	160.197 ns	spdkhc[3]	alarmc	--	--
Total number of failed paths							

Failed Paths
0
0

Slack	Required P2P Time	Actual P2P Time	From	To
N/A	None	160.197 ns	spdkhc[3]	alarmc
N/A	None	160.134 ns	spdkhc[5]	alarmc
N/A	None	160.045 ns	spdkhc[7]	alarmc
N/A	None	160.012 ns	spdkhc[4]	alarmc
N/A	None	159.982 ns	spdkhc[0]	alarmc
N/A	None	159.933 ns	spdkhc[9]	alarmc
N/A	None	159.911 ns	spdkhc[6]	alarmc
N/A	None	159.890 ns	spdkhc[2]	alarmc
N/A	None	159.853 ns	spdkhc[10]	alarmc
N/A	None	159.839 ns	spdkhc[11]	alarmc
N/A	None	159.806 ns	spdkhc[1]	alarmc
N/A	None	159.758 ns	spdkhc[8]	alarmc
N/A	None	159.727 ns	spdkhc[12]	alarmc
N/A	None	159.661 ns	spdkhc[3]	brakec[2]
N/A	None	159.657 ns	spdkhc[3]	brakec[0]
N/A	None	159.648 ns	spdkhc[3]	brakec[1]
N/A	None	159.635 ns	spdkhc[14]	alarmc
N/A	None	159.598 ns	spdkhc[5]	brakec[2]
N/A	None	159.594 ns	spdkhc[5]	brakec[0]
N/A	None	159.585 ns	spdkhc[5]	brakec[1]
N/A	None	159.555 ns	spdkhc[13]	alarmc
N/A	None	159.509 ns	spdkhc[7]	brakec[2]
N/A	None	159.505 ns	spdkhc[7]	brakec[0]
N/A	None	159.496 ns	spdkhc[7]	brakec[1]
N/A	None	159.476 ns	spdkhc[4]	brakec[2]
N/A	None	159.472 ns	spdkhc[4]	brakec[0]
N/A	None	159.463 ns	spdkhc[4]	brakec[1]
N/A	None	159.446 ns	spdkhc[0]	brakec[2]
N/A	None	159.442 ns	spdkhc[0]	brakec[0]
N/A	None	159.433 ns	spdkhc[0]	brakec[1]
N/A	None	159.397 ns	spdkhc[9]	brakec[2]
N/A	None	159.393 ns	spdkhc[9]	brakec[0]
N/A	None	159.384 ns	spdkhc[9]	brakec[1]
N/A	None	159.375 ns	spdkhc[6]	brakec[2]
N/A	None	159.371 ns	spdkhc[6]	brakec[0]
N/A	None	159.362 ns	spdkhc[6]	brakec[1]
N/A	None	159.354 ns	spdkhc[2]	brakec[2]
N/A	None	159.350 ns	spdkhc[2]	brakec[0]
N/A	None	159.341 ns	spdkhc[2]	brakec[1]
N/A	None	159.317 ns	spdkhc[10]	brakec[2]
N/A	None	159.313 ns	spdkhc[10]	brakec[0]
N/A	None	159.304 ns	spdkhc[10]	brakec[1]
N/A	None	159.303 ns	spdkhc[11]	brakec[2]
N/A	None	159.299 ns	spdkhc[11]	brakec[0]
N/A	None	159.290 ns	spdkhc[11]	brakec[1]
N/A	None	159.270 ns	spdkhc[1]	brakec[2]
N/A	None	159.266 ns	spdkhc[1]	brakec[0]
N/A	None	159.257 ns	spdkhc[1]	brakec[1]

N/A	None	159.222 ns	spdkhc[8]	brakec[2]
N/A	None	159.218 ns	spdkhc[8]	brakec[0]
N/A	None	159.209 ns	spdkhc[8]	brakec[1]
N/A	None	159.191 ns	spdkhc[12]	brakec[2]
N/A	None	159.187 ns	spdkhc[12]	brakec[0]
N/A	None	159.178 ns	spdkhc[12]	brakec[1]
N/A	None	159.099 ns	spdkhc[14]	brakec[2]
N/A	None	159.095 ns	spdkhc[14]	brakec[0]
N/A	None	159.086 ns	spdkhc[14]	brakec[1]
N/A	None	159.019 ns	spdkhc[13]	brakec[2]
N/A	None	159.015 ns	spdkhc[13]	brakec[0]
N/A	None	159.006 ns	spdkhc[13]	brakec[1]
N/A	None	158.780 ns	spdkhc[15]	alarmc
N/A	None	158.244 ns	spdkhc[15]	brakec[2]
N/A	None	158.240 ns	spdkhc[15]	brakec[0]
N/A	None	158.231 ns	spdkhc[15]	brakec[1]
N/A	None	111.954 ns	distancec[3]	alarmc
N/A	None	111.908 ns	distancec[2]	alarmc
N/A	None	111.837 ns	distancec[0]	alarmc
N/A	None	111.827 ns	distancec[1]	alarmc
N/A	None	111.698 ns	distancec[5]	alarmc
N/A	None	111.667 ns	resetc	alarmc
N/A	None	111.665 ns	distancec[6]	alarmc
N/A	None	111.628 ns	distancec[7]	alarmc
N/A	None	111.499 ns	distancec[4]	alarmc
N/A	None	111.418 ns	distancec[3]	brakec[2]
N/A	None	111.414 ns	distancec[3]	brakec[0]
N/A	None	111.405 ns	distancec[3]	brakec[1]
N/A	None	111.372 ns	distancec[2]	brakec[2]
N/A	None	111.368 ns	distancec[2]	brakec[0]
N/A	None	111.359 ns	distancec[2]	brakec[1]
N/A	None	111.301 ns	distancec[0]	brakec[2]
N/A	None	111.297 ns	distancec[0]	brakec[0]
N/A	None	111.291 ns	distancec[1]	brakec[2]
N/A	None	111.288 ns	distancec[0]	brakec[1]
N/A	None	111.287 ns	distancec[1]	brakec[0]
N/A	None	111.278 ns	distancec[1]	brakec[1]
N/A	None	111.162 ns	distancec[5]	brakec[2]
N/A	None	111.158 ns	distancec[5]	brakec[0]
N/A	None	111.149 ns	distancec[5]	brakec[1]
N/A	None	111.131 ns	resetc	brakec[2]
N/A	None	111.129 ns	distancec[6]	brakec[2]
N/A	None	111.127 ns	resetc	brakec[0]
N/A	None	111.125 ns	distancec[6]	brakec[0]
N/A	None	111.118 ns	resetc	brakec[1]
N/A	None	111.116 ns	distancec[6]	brakec[1]
N/A	None	111.092 ns	distancec[7]	brakec[2]
N/A	None	111.088 ns	distancec[7]	brakec[0]
N/A	None	111.079 ns	distancec[7]	brakec[1]

N/A	None	110.963 ns	distancec[4]	brakec[2]
N/A	None	110.959 ns	distancec[4]	brakec[0]
N/A	None	110.950 ns	distancec[4]	brakec[1]
N/A	None	110.923 ns	distancec[10]	alarmc
N/A	None	110.815 ns	distancec[8]	alarmc
N/A	None	110.610 ns	distancec[11]	alarmc
N/A	None	110.533 ns	distancec[9]	alarmc
N/A	None	110.514 ns	distancec[13]	alarmc
N/A	None	110.387 ns	distancec[10]	brakec[2]
N/A	None	110.383 ns	distancec[10]	brakec[0]
N/A	None	110.374 ns	distancec[10]	brakec[1]
N/A	None	110.279 ns	distancec[8]	brakec[2]
N/A	None	110.275 ns	distancec[8]	brakec[0]
N/A	None	110.266 ns	distancec[8]	brakec[1]
N/A	None	110.159 ns	distancec[12]	alarmc
N/A	None	110.108 ns	distancec[14]	alarmc
N/A	None	110.074 ns	distancec[11]	brakec[2]
N/A	None	110.070 ns	distancec[11]	brakec[0]
N/A	None	110.061 ns	distancec[11]	brakec[1]
N/A	None	110.044 ns	distancec[15]	alarmc
N/A	None	109.997 ns	distancec[9]	brakec[2]
N/A	None	109.993 ns	distancec[9]	brakec[0]
N/A	None	109.984 ns	distancec[9]	brakec[1]
N/A	None	109.978 ns	distancec[13]	brakec[2]
N/A	None	109.974 ns	distancec[13]	brakec[0]
N/A	None	109.965 ns	distancec[13]	brakec[1]
N/A	None	109.623 ns	distancec[12]	brakec[2]
N/A	None	109.619 ns	distancec[12]	brakec[0]
N/A	None	109.610 ns	distancec[12]	brakec[1]
N/A	None	109.572 ns	distancec[14]	brakec[2]
N/A	None	109.568 ns	distancec[14]	brakec[0]
N/A	None	109.559 ns	distancec[14]	brakec[1]
N/A	None	109.508 ns	distancec[15]	brakec[2]
N/A	None	109.504 ns	distancec[15]	brakec[0]
N/A	None	109.495 ns	distancec[15]	brakec[1]
N/A	None	10.156 ns	driverc	alarmc
N/A	None	9.618 ns	driverc	brakec[2]
N/A	None	9.617 ns	driverc	brakec[0]
N/A	None	9.607 ns	driverc	brakec[1]