UNIVERSITI TEKNOLOGI MARA

CONFERENCE PROCEEDING

ICITSBE 2012

1ST INTERNATIONAL CONFERENCE ON INNOVATION AND TECHNOLOGY FOR SUSTAINABLE BUILT ENVIRONMENT

16 -17 April 2012

Organized by:
Office of Research and Industrial
Community And Alumni Networking
Universiti Teknologi MARA (Perak) Malaysia
www.perak.uitm.edu.my

PAPER CODE: UP 04

PARKING MANAGEMENT SYSTEM USING XILINX FPGA

S.Saliha S. Bahrom[a], N. A. Abdul M.[b] , A.Diyana Rosli[c] and Nurlida Ismail[d]
Universiti Teknologi MARA (Pulau Pinang), Malaysia,
[a] saliha642@ppinang.uitm.edu.my, [b] witchkingx@yahoo.com, [c] anis.diyana@ppinang.uitm.edu.my,
[d]nurlida@ppinang.uitm.edu.my

Abstract
This project is to design and develop a system which has the capabilities to detect empty spaces in the parking lot and indicate the area with the availability of the remaining spaces. This system is applicable in most industries which include the operations of a car parking lot. The main purpose of this project is to upgrade the facilities and technologies in typical car parking system. The system uses FPGA (Field Programmable Gate Array) Chip as the main controller and processor. The system is programmed using VHDL language by implementing a finite state machine (FSM) to control the sequence of the program. The program consists of arithmetic sequence operation and registry to register the availability of the parking spaces. The system will have sensors in each of the parking spaces to detect the vacancies of the spaces. The system will calculate the availability of each parking and produce an arithmetic display using seven segment displays. Each of the parking spaces will also be equipped with a lighted indicator to show the vacancies of the parking spaces for visual searching of the vacant spaces. Hardware implementation of the system is using a Digilent Spartan 3 system board and a sensor circuit that will be interfaced with the system board.

Keywords:FPGA, VHDL, FSM, sensor, seven-segment LED

1. Introduction

In this busy and up-rising world, people tend to move, think and act fast. And so, time is precious and every single second are not to be wasted. Such a thing can be seen in a parking lot, especially in a multi-storey car park where people tend to waste some time searching for a parking space. In a large-scale parking lot, the searching of an available space to park is indispensable. Solution that can implement in a new parking centre or upgrading the remaining parking lot is the objective of this project.

In this system each of the parking lots will be equip with a sensor to detect the present of a car in the lot. An empty or vacant parking lot will be consider as logic '1' and occupied parking lot will be consider as logic '0'. The system will calculate the remaining empty spaces and produce an output of seven segment LED display.

To represent this system, a model of parking system will be made. In this model, the parking spaces will be divided into two zones, A and B with 8 parking lots in each zone. The reason the parking spaces is divided into two zones is to represent the zone as different level of floor or building. Each of the zones also will have a seven segment display to display the remaining parking spaces.

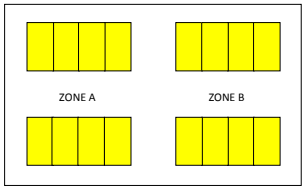

Figure 1.0: Layout of the Parking Lot Model

2. Scope of work

As integrated circuit technology has improved to allow more and more components on a chip, digital system have continued to grow in complexity. As digital systems have become more complex, detailed design of the systems at the gate and flip-flop level has become very tedious and time consuming. For this reason, use of hardware description language in the digital design process continues to grow in importance. [1]
The development of the smart parking system involve in constructing a parking model that will resemble a parking lot, interface circuit and software programming for the FPGA. The most important task in developing is the programmed structure. Xilinx ISE 7.1i was used in writing the RTL (Register Transfer Level) using VHDL language. The language will describe the behavioral of the parking system. A series of simulation were done using ModelSim PE Student Edition 6.4 to produce the output and timing simulation of the code.

2.1 Hardware Design

A model that will be made to represent the system is made from wood. This is to create a miniature parking lot and to demonstrate how the system works. In this model a switch will substitute the used of sensor at each parking space as it is very complicated to install a real sensor to the model. The interface circuit also was design using free software, EAGLE 4.09r2 for windows in creating the PCB layout of the interface circuit. Below is the hardware block diagram.

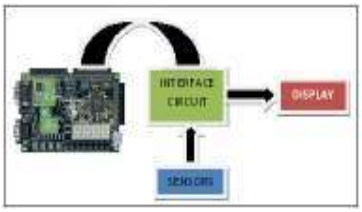

Figure 2.1: Hardware Block Diagram

3. Methodology

3.1 Requirement

The system is design based on counting the number of empty spaces in the parking lot. A method of interpreting the data of the number of parking lot is crucial so that the system will count the exact number of empty spaces and will suit a large number of data interpretation. A creative and simple method is design to make the system less error. Firstly, the system will considered all of the parking lot and will represent an array of binary numbers. The binary number will represent a data that will be use to count the parking lot. The system will understand that bit '0' of the data is considered as empty and bit '1' as occupied. The next step of the system is to count the only bit '0' in the data. The system will detect the less significant bit of the binary numbers. When the less significant bit of the data is desired the counting begins. If the desired bit is not detected the counting process is not done. The next step is shifting the data to the right so that the system will count the bit next to the less most significant bit before. The whole shifting and counting in done continuously until the whole data is interpreted. Below is a representation how the counting and shifting process occurred.

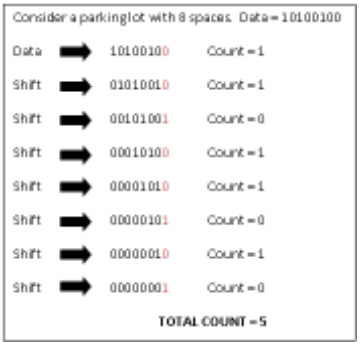

Figure 3.0: Counting and shifting process

3.2 Top level architecture

The whole system can be considered as a combination of three main blocks. The first block, the sensors block is the outer block because it contains sensors that located at each of the parking lot. The second block is the counter block. This is where all the counting and shifting operation occurred. This block is the processor of the system that lies in the FPGA chip. The last block is the display block. This block will have seven segments display and light indicator for driver viewed. Figure 3.1 below is the top level architecture of the system.

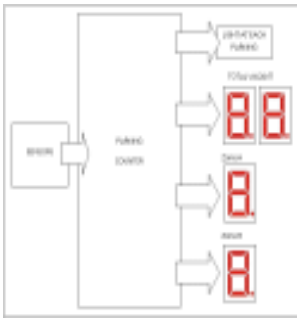

Figure 3.1: Top level architecture of the system

Figure 3.2 below is the top level symbol of the system. Here details of the system inputs and outputs are listed below.

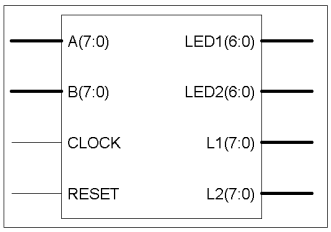

Figure 3.2: Top level symbol of the system

INPUTS:

A [7:0]	→	The parking data input of zone	A that synchronously with the rising edge of the clock
B [7:0]	→	The parking data input of zone	B that synchronously with the rising edge of the clock
Clock	→	The clock input to the system.	
Reset	→	Reset is asynchronous and	reset the system to initial state.

OUTPUTS:

LED1 [6:0]	→	Display the number of empty	spaces in zone A.
LED2 [6:0]	→	Display the number of empty	spaces in zone B.
L1 [7:0]	→	The light indicator for zone A.	
L2 [7:0]	→	The light indicator for zone B.	

3.3 Design flow

3.3.1 VHDL code

VHDL code is the writing model of this system. The code will describe the behavioral of the system as well as used for simulation. The code can be break into a few segment. The first segment is to declare the entity of the code. This is to declare the input and output signals of the module.

An entity specifies the input and output a signal of a circuit, but it does not give any details as to what the circuit represents. [2] So architecture is to describe the functionality of the circuit. In this system, the architecture can be separate into two. The first part is to define the next state function. The next state function is to give structural of the state machine in this system. Figure 3.3 at the next column is the next state function.

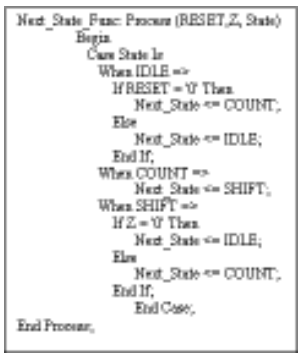

Figure 3.3: The Next state function

The second part of the architecture is the datapath function. The datapath function is to describe the process that will occur at each state. The state involves according to the next state function will perform the operation on counting and shifting. Figure 3.4 below is the datapath function. The datapath function also will set the input A and B into Q register. The C registers also were set for the process of counting.

Figure 3.4: The Datapath function

Another bottom level of function is the decoder. The decoder will be used to decode the TOTAL register for displaying the number of empty parking spaces. The decoder is created into another module because the register will use the same type of decoder to display the data.

3.3.2 Synthesize

The VHDL codes were written and synthesize using Xilinx 7.1i. The purpose of synthesizing VHDL code is to convert it into a net list. This net list will be used to map, route and download into FPGA chip located on the Digilent Spartan 3 system board. [3]

Synthesizing can check for error, number of state machine involves, how many registers, flip-flop and arithmetic module were used for the system. The number of IOB and slices used also can be known.cSynthesize is important so that designer can accommodate the used of the slices and IOB in FPGA according to the design.

4. Result

4.1 Synthesize report

The synthesize report is used to analyze the synthesize VHDL code. A real circuit can be generate from the code as HDL is a hardware description language and the number of state, component and timing of the circuit can be determine. The circuit will not be shown in this report as it is not necessary for explanation. Only the numbers of components that will be used in the circuit is documented.

From the HDL synthesis report:

```
Macro Statistics
# FSMs                      : 1
# ROMs                      : 2
 16x7-bit ROM               : 2
# Adders/Subtractors                : 2
 4-bit subtractor           : 2
# Registers                 : 22
 1-bit register             : 18
 4-bit register             : 4
```

One state machine is used in this system. Two 16x7-bit ROM is used as there is a decoder that will be used for both seven segment display components. Two 4-bit subtractor will be used as there are two register that will be used for subtraction. And twenty-two registers for all the C, Q and TOTAL register.

After the HDL synthesis, there is another extracted report that is the number of device utilization summary. This summary is to show the number of device utilization for 3s200ft256-5 Xilinx FPGA.
From the summary:

```
# of Slices: 36 out of 1920          1%
# of Slice Flip Flops: 35 out of 3840 0%
# of 4 input LUTs: 67 out of 3840     1%
# of bonded IOBs: 48 out of 173       27%
# of GCLKs: 1 out of 8                12%
```

From the summary, the device only use 35 flip-flops and that are about 0.91% of the device utilization, 4 Look Up Table (LUT) about 1% and 48 bonded IOB about 27% of the device utilization. The system required a clock signal and it has use 1 of the 8 clock provided in the Spartan 3 system board. The timing analysis of the system also can obtain from the synthesis.

```
Minimum period:
5.445ns (Maximum Frequency: 183.660MHz)
Minimum input arrival time before clock: 2.514ns
Maximum output required time after clock: 7.896ns
Maximum combinational path delay:
7.555ns
```

4.2 Simulation Result

Figure 4.0 below is the timing diagram from ModelSim PE Student Edition 6.4. This simulation is done by compiling the VHDL code in the simulation software. Here a predetermine input of 01001100 that will be equal to 5 empty spaces were set for input A and 10011101 that is equal to 3 empty spaces were also set for input B.

From the result, at 200ns the first transition of state IDEAL to state COUNT is occurred. Here the CB register will perform a subtraction of 1 from the 1000. This is because the less significant bit of QB contains the bit 1. And the state will move to from state COUNT to state SHIFT. Here both the Q registers will shift a bit from left to right. At 600ns both the C register will perform a subtraction of 1.

The system will perform continues operations of count and shift until the entire bit in both Q registers will be 0. At 1900ns the TOTAL registers will have the final result of 5 empty spaces and 3 from input A and 3 empty spaces from input B. From the simulation result the system has followed the rule to calculate the empty spaces in each of zone.

Figure 4.0 below is the timing diagram from ModelSim

The system uses FPGA device as its main processor and controller. From the synthesis report, the utilization of the device is important so that the system that was written in VHDL code does not utilize more than the device

can supported. A complete care of writing by not using more commands and complicated operation is recommended. This is to ensure that the device can support the system operation.

From the simulation, the timing required for a complete operation of counting the total number of parking is about 2000ns for this model of 16 parking lots before the actual output number is displayed at the seven segment display. The time taken to produce an output from each input is not quite long because for human eyesight, the change in display is very fast. The system can change the number of parking at is desirable. But the time taken is longer for each time the number of parking is increased. But there is a way to counter this limitation. Divide or assign the number of parking into several zones as the process of counting is done in parallel and simultaneously. By dividing the number of parking into several zones, more display of each zone can ease the driver to find empty spaces as the number of searching can be narrowed down.

5. Conclusion

A smart parking system is design to ease the operation of parking a car. There are many types of parking system that can be utilized but there also factors that need to be considered. The most important thing in designing a parking system is maximizing the used of parking and the ease of a driver to park a car. This system is best to implement at a large number of parking spaces because the number of parking can be maximized without a driver missed a parking space because each the zone in parking lot is equipped with a display for empty spaces. The second factor is that user can ease the operation of parking a car as the driver will know which zone that is full and instantly know the empty spaces from the light indicator at each lot. Another advantage of applying this system is that the time taken for a person and the traffic when a lot of user trying to park at the same time can be reduced as driver will know the direction to park. The cost factor to apply this system is rather much less expensive as the system is much cheaper than autonomous system that required railing system, computerized mechanical system for automatic parking and completely restructure the parking building or lot. In conclusions, a smart way to design a system is not just simply by how high technology and complicated the system used but how the system solved, reduce and ease the cost, time and operation for both the user and operator of the parking lot.

References

Charles H. Roth, Jr "*Digital systems design using VHDL",* PWS Publishing Company 20 Park, Plaza, Boston, MA 02116-4324

Stephen Brown, Zvonko Vranesic, "Fundementals of Digital Logic with VHDL Design" Second edition, 2005, McGraw Hill.

David M. Sendek, "Xilinx Project Navigator Reference Guide", 31July2003, PDF

Pinout and Area Constraint Editor Overview, http://toolbox.xilinx.com/docsan/xilinx10/help/iseguide/mergedProjects/pace/pace.htm