International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-2 Issue-4, April 2013

Implementation of ATM Algorithm through VHDL

Kuldeep B. Shukla, Hetal N. Rao, Arjun H. Joshi

Abstract— ATM (automatic teller machine) is a very essential tool required for the society in order to facilitate the need of safe transaction of money. Using this facility one can easily perform the various functions such as balance inquiry, withdrawn, money transfer etc. As this machine operation rely on bank cards, proper password, enough amount of money in one's account, certain verification and identification methods etc. It needs to be secure and having integrity of fine level right to its coding stage for optimum utilization of the service. In order to meet such requirements the coding languages used for it are modified here. The conventional coding styles using 'C' and/or 'C++' are replaced by the VHDL code language so that the attacker cannot easily crack the security levels. In this article, the code composed of VHDL language is suggested for this purpose of security.

Index Terms— Automatic Teller Machine (ATM), VHDL, Krypton Board, Integrity, Security Level.

I. INTRODUCTION

This document contains concept of using VHDL code used in Automatic Teller Machine (ATM) instead of conventionally used codes composed of higher level language. The hacker can easily crack such codes as these all are user friendly languages and therefore it is as easy to interrupt as to compose. Therefore, ATM can be made operating on coding based on Very High Speed Integrated Circuit Hardware Description Language (VHDL). The code made of this language is comparatively hard to crack and modify in order to perform any misuse or hijacking of the original functioning of ATM.

The ATM transaction contains the following three main stages while operating: (1) *Card Authentication* in which card details are read and authenticated by the ATM and (Point Of Sale) POS terminal. (2) *Cardholder Verification* in which the person who inserts the card is verified either by PIN or Signature. (3) *Transaction Authorization* in which the issuing bank decides whether the transaction should be further precede or not [2].

II. SOLVING CONVENTIONAL CODE PROBLEM USING VHDL

The problem arising using the codes other than VHDL code i.e. 'C' and 'C++', is that they can be modified and corrupted using very less efforts and low level of complications. Also, the conventional codes are simulated through a numbers of blocks and after that it is implemented at the ATM machine.

Therefore, it can be interrupted at any stage. This makes the code less secure and easy to crack/modify.

VLSI based programming language i.e. Very High Speed Integrated Circuit Hardware Description Language (VHDL) is also one of the programming language, used for various applications of real time world.

Using this VHDL code, any person who wants to interfere the code, it provides comparatively immune platform of complexity and a more secure version over disclosure of the originally functioning software programming. As it simulates through no such more stages, and can be directly implemented to the machine, it is too worse for the attacker to track the code.

III. VHDL CODE IMPLEMENTATION

Here, we have generated and suggested the flow chart (See Fig.1) of the same functioning program using VHDL programming language, which is analogous to the generally used other programming languages. As per stated earlier, it is having certain benefits over the existing operating codes.

In ATM cards, it executes on EVM (Electro Voting Machine) protocol which precedes the selected transaction that is encrypted with cryptographic MAC (Message Authentication Code) using symmetric key which is shared with the bank that issues the card and it is desired that the bank must be capable to detect such code in any condition. The whole process can be described as following: (1) The customer/card holder first inserts the card at the ATM machine at the space provided for card insert. In this stage, it is verified whether the inserted card is valid or not. If it seems to be invalid, then it is indicated though the buzzer. If it is the valid one, then it further proceeds. As the next step it asks for a valid password to be inserted by the card holder. Again, if the entered password is wrong, a next chance is provided to the user to re-insert the correct password. In case of the entered password is correct at the first attempt, the system allows to proceed further. (2) In this state, now after inserting the correct password, it asks for the language selection having two or more options from which any one can be selected by the user. After select the main menu which contains the various functioning areas to be selected. Here, in this case, we have taken three options after this stage.

They are, (a) saving account, (b) check balance, (c) current account.

Manuscript published on 30 April 2013.
* Correspondence Author (s)
Kuldeep B. Shukla (PG Student), Electronics and Communication, SAL-ITER, Ahmedabad, Gujarat, India.
Hetal N. Rao (PG Student), Electronics and Communication, RK University, Rajkot, Gujarat, India.
Arjun H. Joshi (Asst. Prof.), Electronics and Communication, SAL-ITER, Ahmedabad, Gujarat, India.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license http://creativecommons.org/licenses/by-nc-nd/4.0/

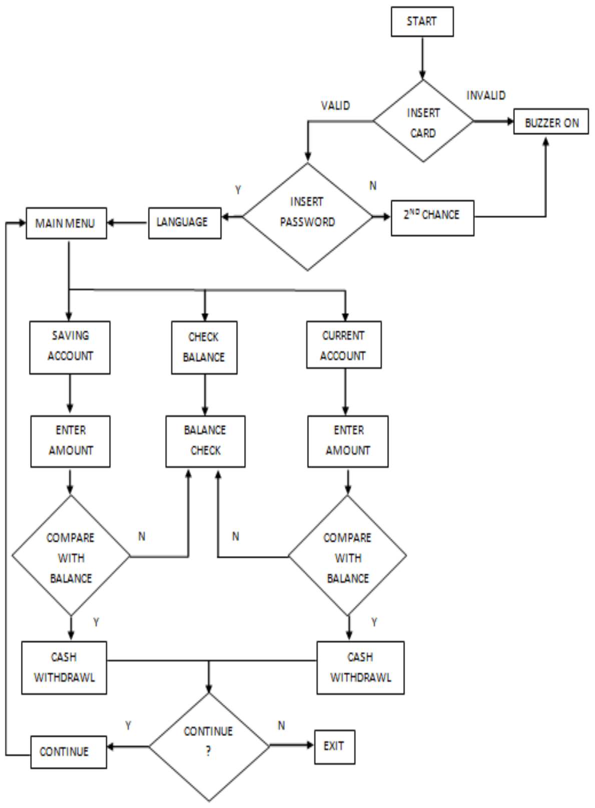


Fig.1 Flowchart of VHDL code of ATM machine

By selecting the specific option from this list, the corresponding steps are preceded. (a) Consider, option *saving account* is selected. Then it asks for amount of money to be inserted that are to be withdrawn. Then this amount is compared with the total balance present in the account. (b) If this option is selected, then it just compares with current amount of the money present in the account and is displayed on the computer screen. If option (c) is selected then, the same procedure is carried out as in option (a) but with considering the *current account.* In both the cases of (a) and (c), the demanded amount of money are first compared with the present balance of the corresponding accounts, if the demand *withdrawn* takes place. Otherwise, in case of the amount more than the present balance, it denies the access. (3) In this phase, the system asks the user to further carry on/continue for the next transaction or not. If the user wants to continue, then the whole process right from the state (1) repeats and if no further transaction is required i.e. user selects *No* as answer of the question *continue for the next transaction ?*, it exits. Thus, the whole process shown though the above flow chart takes place practically.

IV. SIMULATION TOOLS

Spartan FPGA:

In performing this code, the hardware Krypton Board (Version 1.1) of Spartan 3E is used. This board contains the specifications as given now. It is having the *lead time* of 2 weeks. It provides the designer a faster access. It includes the various components as power supply (100-240 V, 50-60 Hz), development board, evaluation software, USB cable etc. It is the advanced version of the Spartan FPGA tool kit. It contains the additional features such as 128 M-Bit Parallel Flash Memory, 16 M-Bit SPI Flash Memory and 64 M-Byte DDR SDRAM. The operating clock used is generated by 50 MHz crystal oscillator. Consider the development board of ig. [3].

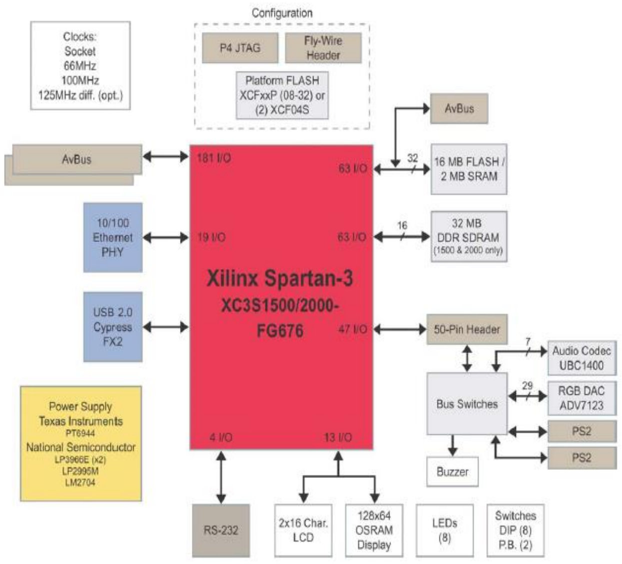


Fig.3 Spartan-3 Development Block Board Diagram [5]

The functioning of it can be seen easily from the above diagram on this hardware using VHDL simulator Xilinx 9.2i. While implementing practically, first the power supply and clocks are to be considered. In this case, the board is having a variety of clocks and the operating voltages are of range 12-18 V. From the available onboard memories, we have utilized 16 MB FLASH/2 MB SRAM for the purpose of execute the code of VHDL. RS232 serial port is used for serial data fetch. JTAG port is used as debug interface which is utilized for downloading the executable file to the external/internal memory. Avnet Std.bus is used as the I/O buses. The operating voltages must be of range 12-16 V .The inputs are given by the switches as DIP or Push Button switches. The corresponding output to the loaded VHDL code from the Xilinx software is displayed in form of LCD or LED display that operate on the digital inputs (i.e. 0 and 1). The buzzer mentioned in the VHDL code is carried out practically by using the buzzer of the board itself. Also we can use LEDs for the purpose of buzzer.

The VHDL code is implemented over this hardware tool and the results are gathered and compared with the simulation results obtained by the simulation tools. The corresponding simulation results are mentioned in the next sections.

Here, below table indicates the different no of software port names which designate the performance of input. Then port type is the behavior of input and performance in the given algorithm. And last column specify how the blocks performed.

Table I is having seventeen different types of performing pin in which one is behaves like input one is output and remaining ports are consider as a intermediate blocks.

TABLE I. PORT DESCRIPTION

No.	*Port Name*	*Port Type*	*Port Description*
1.	InsertCard	Input	Asks to insert the ATM card and checks the validity of it.
2.	Buzzer	Output	Indicates the result of Validity Check.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

No.	***Port Name***	***Port Type***	***Port Description***
3.	EnterPassword	Inout	Asks to enter the password iff the card is found Valid.
4.	SecondChance	Inout	Provides second chance in case of wrong Password entered.
5.	Language	Inout	Provides the selection of language to proceed further.
6.	SavinngAccount	Inout	Asks user to select it if he/she wants to concern with *saving account*.
7.	CurrentAccount	Inout	Asks user to select it if he/she wants to concern with *current account*.
8.	CheckBalance	Inout	Asks the user if he/she wants to check the *Balance*
9.	EnterAmountSA	Inout	To enter the amount of money to be withdrawn from *Saving Account*.
10.	EnterAmountCA	Inout	To enter the amount of money to be withdrawn from *Current Account*.
11.	BalanceCheck	Inout	Compares the entered amount of money with the available balance of the corresponding account.
12.	Balance	Inout	It indicates the net amount
13.	MoneyWithdrawlSA	Output	As an output, it returns the entered amount of money from the *Saving Account* to the user after completing all the tests and checks.
14.	MoneyWithdrawlCA	Output	As an output, it returns the entered amount of money from the *Current Account* to the user after completing all the tests and checks.
15.	Continue	Inout	Allows the user to continue the transaction process again by choosing *'YES'* option.
16.	Depart	Output	Allows the user to exit the transaction process after completing transaction process once, by choosing *'NO'* option.
17.	Invalid	Inout	Activates the *Buzzer* if the ATM Card is found *invalid.* and 'Invalid' option becomes Active/High.

Using the software, Xilinx 9.2i ISE the implementation of the ATM code is performed practically. The FPGA hardware used is Spartan 3 series on which we have implemented and practically performance of the ATM code is carried out successfully. The simulation results are presented in the next section with the snapshots of the hardware implementation we have carried out.

V. SIMULATION RESULTS

This obtained RTL Schematic Diagrams are given in fig.4. VHDL code, which we have prepared, is implemented in the software and the corresponding RTL views are obtained as per shown in fig.4 and fig.5. The RTL Technology diagram is also given in fig.6 which is obtained by simulating the ATM machine VHDL code in the Xilinx 9.2i software.

In the fig.7, the console diagram is shown that we obtained as a result of simulation of the ATM code using VHDL. Table II indicates simulation port description.

TABLE II. SIMULATION PORT DESCRIPTION

No.	***Port Name***	***Status***	***Function Description***
1.	InsertCard	1	As the ATM card is inserted, this port is High.
2.	Buzzer	0	The card is Valid it is disable.
3.	EnterPassword	U	The entered password must seem not visible, this port status remains **undefined**.
4.	SecondChance	U	It is the first chance. So, **undefined**.
5.	Language	U	Once, language has been selected, it becomes **undefined**.
6.	SavinngAccount	1	If the *saving account* has been selected. So, port status is1.
7.	CurrentAccount	1	If the *current account* has been selected. So, port status is1.
8.	CheckBalance	U	As the option check *Balance* is not selected, it is **undefined**.
9.	EnterAmountSA	U	To enter the amount of money to be withdrawn from Saving Account.
10.	EnterAmountCA	U	To enter the amount of money to be withdrawn from Current Account.
11.	BalanceCheck	0	This option is not selected then the status is zero.
12.	Balance	U	As an account has been selected, the status of net balance should be invisible. Hence the status is **undefined**.
13.	MoneyWithdrawlSA	U	It becomes one when money transaction is from *saving account*. As the with-drawl has not been done it is **undefined** *(now)*.
14.	MoneyWithdrawlCA	U	It becomes one when money transaction is from *current account*. As the with-drawl has not been done it is **undefined** *(now)*.
15.	Continue	U	It will become high if the user wants to repeat the whole process. Right now no decision of continue or depart has been taken it is **undefined**.
16.	Depart	U	It will become one if the user exits the process. Yet decision is not taken, Hence **undefined**.
17.	Invalid	U	The card valid therefore the statue of this port is **undefined**. If the ATM card since invalid then the status will be one.

VI. RESULTANT ISSUES

Some of the issues arising while performing and generating this code for ATM using VHDL are as given below:

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

i). The size of vector that we can simultaneously use is limited up to only 10 vectors at a time using Spartan FPGA kit when we are implementing the code practically. But we require this size of vectors to be used simultaneously is 19 (i.e. we have to use vector size equals to 0 to 18 or 18 down to 0).

ii). The another such an issue is that we can also add certain security levels in this code so that the coding cannot be altered by any other person up to certain amount of possibility. These are the issues that we faced during performance and implementation of the VHDL based ATM algorithm. Practically that can lead to the further research.

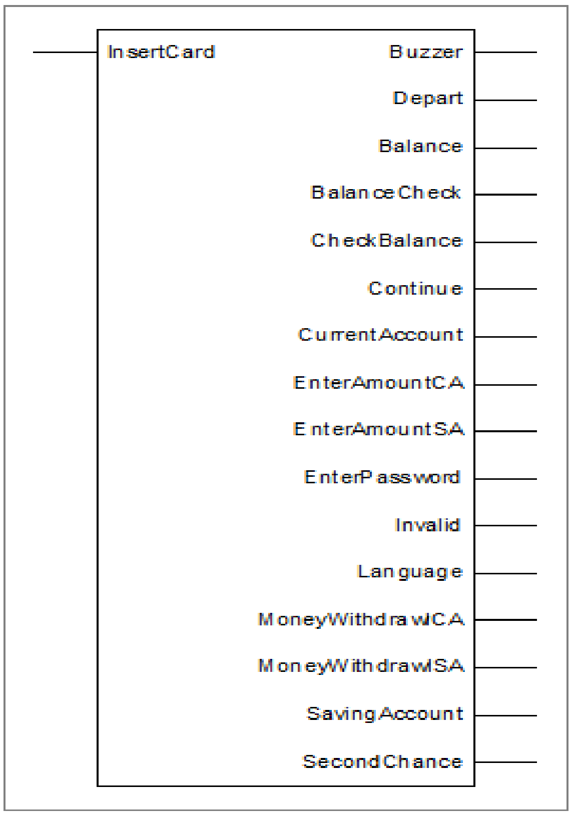


Fig.4 RTL Schematic Diagram-1 of Xilinx 9.2i ISE

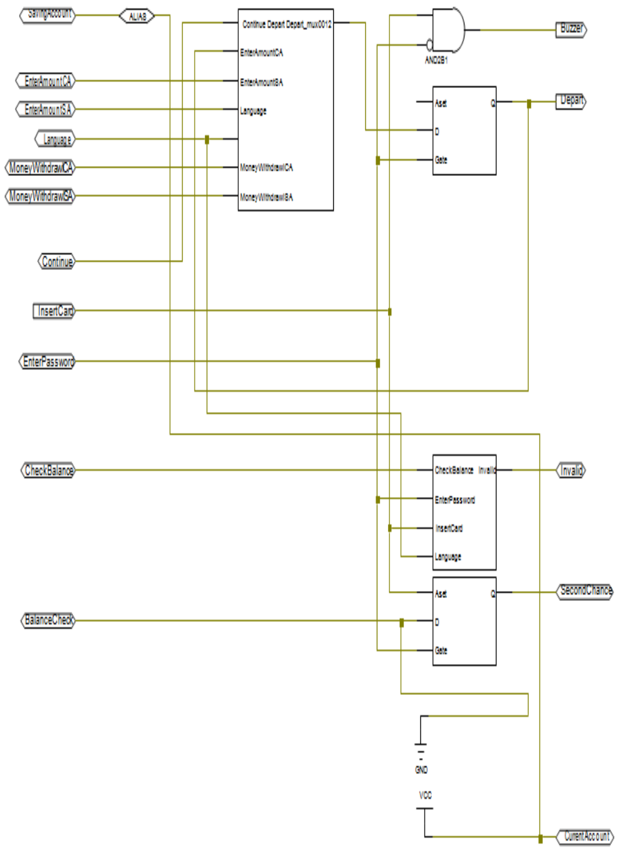


Fig.5 RTL Schematic Diagram-2 of Xilinx 9.2i ISE

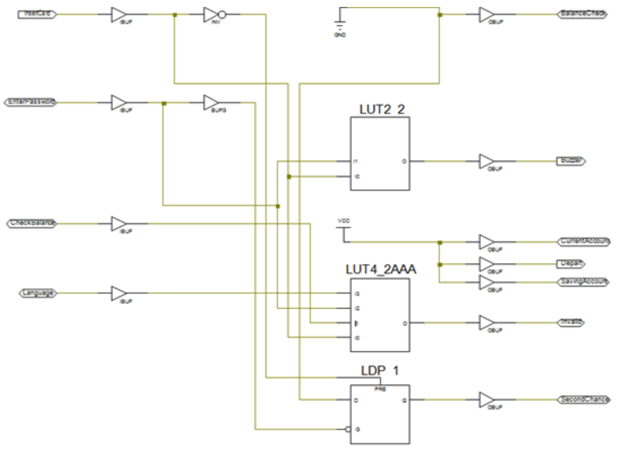
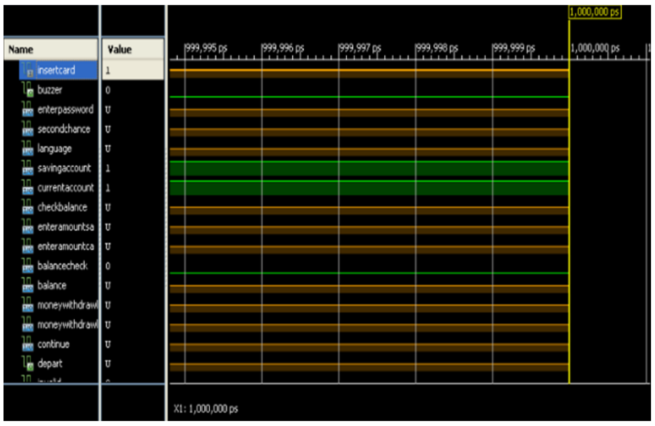


Fig.6 RTL Technology Diagram of Xilinx 9.2i ISE



```
Name            Value
insertcard      1
buzzer          0
enterpassword   U
secondchance    U
language        U
savingaccount   1
currentaccount  1
checkbalance    U
enteramountsa   U
enteramountca   U
balancecheck    0
balance         U
moneywithdrawl  U
moneywithdrawl  U
continue        U
depart          U
```

Fig.7 Simulation Console Diagram of Xilinx 9.2i ISE

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

VII. CONCLUSION

The replacement of ATM machine algorithm by the VHDL code leads to a level of security for keeping integrity of code as well as original functioning.

The algorithm can be further modified for the purpose of higher security than that of conventional higher level language code which prevents easy cracking of the code so that enough time can be obtained to track the bug (hacker) that spoils the proper functioning of the ATM software.

REFERENCES

[1] Yingxu Wang and Yanan Zhang, “The Formal Design Model of an Automatic Teller Machine (ATM)” University of Calgary, Canada, International Journal of Software Science and Computational Intelligence, 2(1), 102-131, January-March 2010.

[2] Mike Bond, Omar Choudary, Steven J. Murdoch, Sergei Skorobogatov, and Ross Anderson, “Chip and Skim: cloning EMV cards with the pre-play attack”. Computer Laboratory, University of Cambridge, UK.

[3] Beginner’s Guide to Xilinx Spartan-3E FPGA Starter Kit Board (Revision 2), By Nick Desaulniers and Cody Cziesler, RIT Department of Computer Engineering, 5/5/11.

[4] Pong P. Chu "RTL Hardware Design Using VHDL, Coding For Efficiency, portability and Scalability", Willy Interscience a john wilay & sons, inc., Publication, pp no:23-160.

[5] Avenet Avenue, user’s guide, Xilinx® Spartan™-3 Development Kit.

[6] Stanley MAZOR and Patricia LINGSTRAAT, "A guide to VHDL (2nd Edition)", copyright 1993, Kluwer Achedamic Publishers, pp no: 1-1 to 7-16.

[7] Peter J. Ashenden and Jim Lewis, "The Designer's Guide to VHDL (3rd Edition)" copyright 2008, Morgan KJaufmann Publication, pp no: 207-225.

[8] http://communities.mentor.com/mgcx/servlet/JiveServlet/previewBody/2324-102-2-5693/HDL-Standards-UG-CAST.rev1b.pdf.

[9] http://www.fpga.com.cn/hdl/training/Vhdl_Golden_Reference_Guide.pdf

*Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)
© Copyright: All rights reserved.*