Acta Universitatis Sapientiae
Electrical and Mechanical Engineering, 1 (2009) 93-104

FPGA Implementation of Fuzzy Controllers and Simulation Based on a Fuzzy Controlled Mobile Robot

Sándor Tihamér BRASSAI

Department of Electrical Engineering, Faculty of Technical and Human Sciences,
Sapientia University, Tîrgu Mureş, Romania,
e-mail: tiha@ms.sapientia.ro

Manuscript received March 15, 2009; revised June 30, 2009.

Abstract: Fuzzy controllers offer a simple solution to the problem of controlling a system. Combining the simplicity of fuzzy systems with the parallel implementation on FPGA circuits, a very fast controller can be obtained. This paper presents a fuzzy controller implemented on an FPGA circuit, with membership functions, rule table, inference system and defuzzification modules implemented on this hardware tool. In the final part of the paper the control of a mobile robot with a fuzzy controller is exemplified.

Keywords: FPGA, hardware implementation, fuzzy control, mobile robot.

1. General information

The main purpose of the paper is presentation and practical application of a fuzzy controller implemented on a digital reconfigurable hardware (FPGA). There are several methods for implementation of fuzzy controllers or other discrete controllers, such as implementation on PC, microcontrollers, DSP digital signal processors, DSC digital signal controllers, PLC, on ASIC circuits or FPGAs. In this paper, a fuzzy controller implemented on FPGA is discussed. Some of the FPGA implementation advantages are: parallelization possibility, the advantages resulting from reconfigurability (scalable size of the fuzzy processor) and very high speed operation.

For the description of the controller's structure, VHDL hardware [1], [2], [3] description language was used with XILINX ISE development software. As a target device, a NEXYS-2 development board with SPARTAN3E 1200 FPGA chip is proposed. In the paper, a short overview of the components of a fuzzy controller (fuzzification, defuzzification, inference system, rule table [4], [5]) as well as the FPGA-based implementation of these components are described. The controller system is proposed to be used for a mobile robot track-following problem.

In the first step of the controller design, the controller applicability was tested for the mentioned track-following problem by using a model for the robot simulation, which will also be discussed. The robot model will also be presented in the work. The results of the simulation are described. The paper ends with the conclusions and future ideas.

2. The control structure and the robot model

For the control of a mobile robot, a fuzzy controller was chosen because it is an efficient method for system control. In case of the fuzzy controller, the control strategy can be described based on simple rules such as (the examples will be given for the mobile robot):

- *if* the target point is on the right hand side in front of the robot*, then* the robot has to move forward rotating to the right
- *if* the target point for the robot is behind, *then* the robot has to turn back (see *Fig. 1.*)

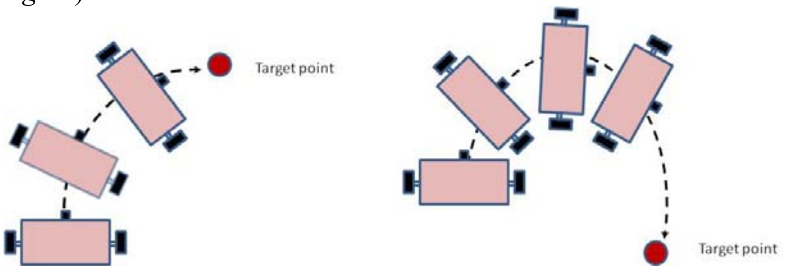

Figure 1: Rule exemplification.

The main objective is to control the robot to follow a predefined target trajectory. A robot with three wheels has been used. The left and right side wheels are driven by DC motors, the third wheel is a free wheel. The robot model used in simulations is presented in the following equations [6].

$$x[k+1] = x[k] + L\frac{v2+v1}{\xi + |v1-v2|}\sin(\varphi[k]) \tag{1}$$

$$y[k+1] = y[k] + L\frac{v2+v1}{\xi + |v1-v2|}\cos(\varphi[k]) \tag{2}$$

$$\varphi[k+1]=\varphi[k]+\frac{v2-v1}{2L}T \tag{3}$$

In *Fig. 2.* the XY coordinate system in which the robot's position is measured and the X_RY_R coordinate system attached to the robot are shown, where:

(x_r,y_r) - the robot's position in the main XY coordinate system;

φ -the robot's orientation in the XY coordinate system;

L - the distance between the robot's side wheels;

θ_r - the next target point orientation measured in the X_RY_R coordinate system attached to the robot.

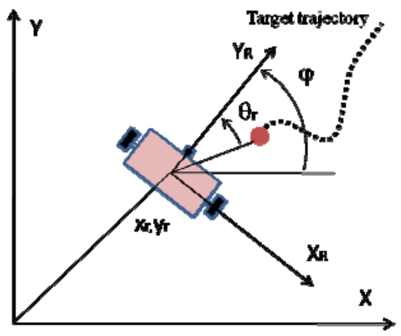

Figure 2: System variables specifications.

The structure of a fuzzy controller is presented in *Fig. 3*. One of the two inputs of the controller is the distance between the origin of the robot coordinate system (X_RY_R) and the target point. The second input is the robot orientation (θ_r). The two output signals of the controller represent the control signals for the two DC motors. The DC motors are controlled by pulse modulation with a pair of H bridges. A FUJITSU based DICE-KIT development board with 16 bit MB90F352 microcontroller is used for the PWM signal generation. [7], [8].

3. Fuzzy controller structure

The Fuzzy controller structure with fuzzification, defuzzification, inference system and rule base module is presented in *Fig. 3*.

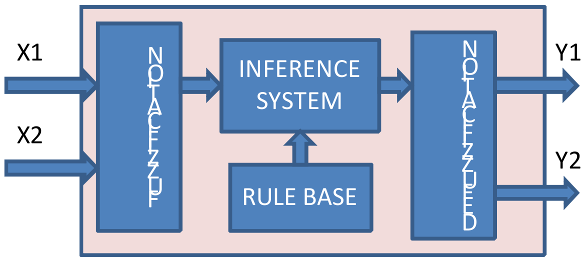

Figure 3: Fuzzy controller structure.

During fuzzification the inputs are converted into fuzzy variables (linguistic variables). The input spaces and the output universes are covered with membership functions (MF). Using the fuzzification procedure for an input within the input universe, based on fuzzy set theory, a membership value will result for each active MF. The membership values are connected to the outputs through the inference system and the rule-base table. The rule-base table defines which output MF will be selected for a pair of active input MFs. As it can be seen in *Fig. 4.*, two MFs are selected for both input spaces. The *a*, *b* and *c* vector variables represent the MF definition parameters.

The fuzzy rule base is composed by IF-THEN statements. The IF-part of the rule represents the antecedence and composes the input conditions. The THEN-part represents the consequence of the rule [4].

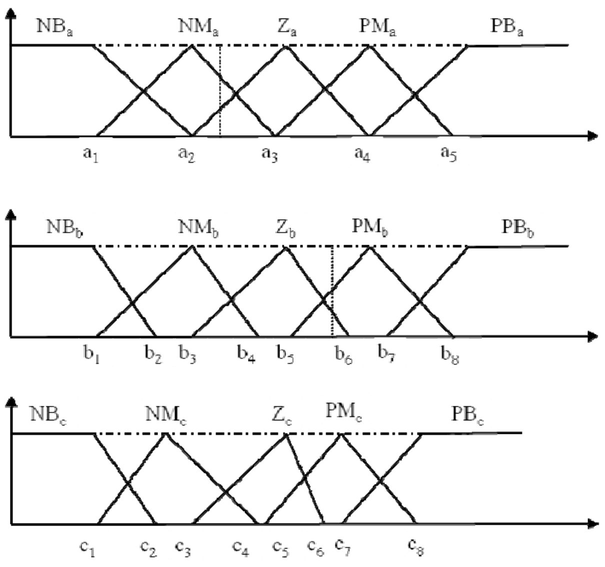
NB_a
NM_a
Z_a
PM_a
PB_a
a_1 a_2 a_3 a_4 a_5
NB_b
NM_b
Z_b
PM_b
PB_b
b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8
NB_c
NM_c
Z_c
PM_c
PB_c
c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8

Figure 4: Input and output universes membership functions.

The fuzzy rule sets usually have as many antecedents as the number of input variables and as many consequences as the number of output variables, these are described using fuzzy IF-THEN statements:

$$\text{IF } x_1{=}A_i \text{ and } x_2{=}B_j \text{ THEN } y{=}C_k. \tag{4}$$

An input-output relation is described with a 3 variable fuzzy rule, where x_1 and x_2 represent the inputs and y the output.

For the antecedent part are used the:

$$A_i \quad \mu_i : X_i \rightarrow [0,1];\ i = 1 \cdots n_1,\ B_j \quad \mu_j : X_j \rightarrow [0,1];\ j = 1 \cdots n_2 \tag{5}$$

fuzzy sets, respectively for the consequence part the

$$C_k \quad \mu_k : X_k \rightarrow [0,1];\ k = 1 \cdots n_3 \text{ fuzzy set.} \tag{6}$$

The number of membership functions for the two inputs are n_1 and n_2, and for the output variable is n_3.

If for each of the two inputs two MFs can be activated, then looking to the rule base table (based on (4)), maximum four rules can be considered. For the ‘and’ logic operator the ‘min’ function is considered, and for the ‘or’ logic operator the ‘max’ function is used.

An example of the active rules of the mobile robot, is in *Fig 5*.

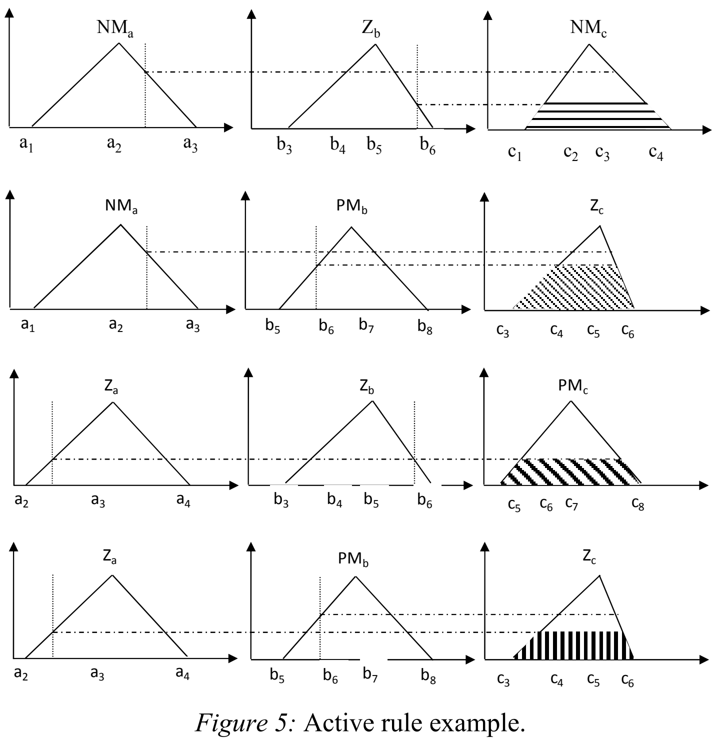

Figure 5: Active rule example.

The result for a fuzzy inference system is a fuzzy set (*Fig. 6.*). To compute the output value, it is necessary to evaluate, defuzzificate the resulted fuzzy set. For defuzzification, a large number of defuzzification algorithms can be used. In this paper the center of gravity (COG) defuzzification algorithm is used.

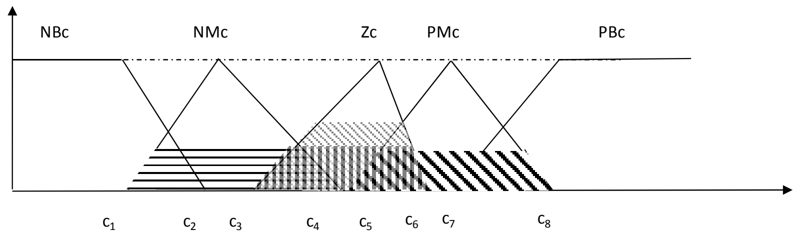

Figure 6: The resulting fuzzy sets.

4. Hardware implementation of a fuzzy controller

A. Membership functions implementation

In the project, triangle shaped MFs are used. The MFs are stored in BRAM memories. For each MF the function form and other parameters, such as the displacement distance of the MF measured to the input space origin, the center of MF, the length of the MF and the BRAM starting address for function form storage are stored. A memory composed of multiple BRAMs is used for function form storage and another BRAM memory is used for parameter storage. In *Table 1* and *Fig. 7.* the MF's stored parameters are presented.

Table 1: Membership function parameters stored in BRAM

offset	length	center	storing address

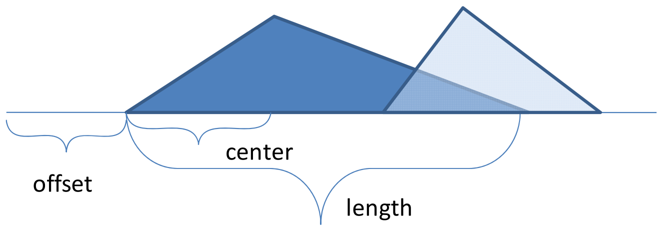

Figure 7: Membership function parameter specifications.

B. Rule-base table implementation

For a system with two inputs a number of maximum four rules are active for an input pair.

The rules are stored in a BRAM memory. The rule of BRAM addressing is composed based on a pair (A, B) of input MF indexes stored in index registers. In the rule table, the indexes of the MFs from the output universe are stored.

C. Simplified steps for the fuzzy controller

In *Fig. 8.*, the data path for controller output processing for an input data pair is presented. The controller output process is composed of the following steps:

a.) Membership value calculation and the active MF index computing. With the MFs offsets and length, the active MF index and member value are determined and stored in index and member value registers. The variables i, j, k, l represent the indexes and $\mu_{i,}$, μ_j, μ_k, μ_l represent the membership value registers.

b.) In step two the rule memory is indexed.

c.) The active output MFs index is extracted.

d.) Based on output MF indexes, these parameters are extracted from the output MF parameter storage BRAM and stored in output parameter registers with alpha cut results.

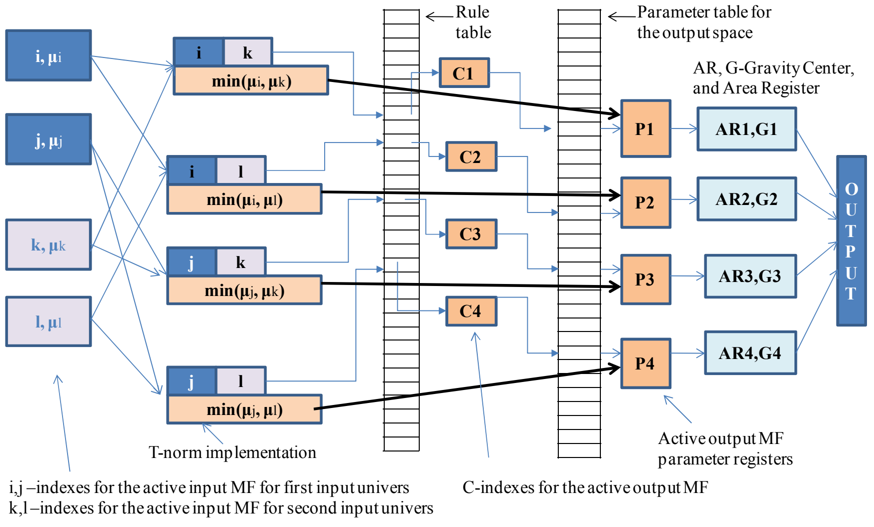
Rule table
Parameter table for the output space
AR, G-Gravity Center, and Area Register
i, μi
j, μj
k, μk
l, μl
i k
min(μi, μk)
i l
min(μi, μl)
j k
min(μj, μk)
j l
min(μj, μl)
C1
C2
C3
C4
P1
P2
P3
P4
AR1,G1
AR2,G2
AR3,G3
AR4,G4
OUTPUT
T-norm implementation
Active output MF parameter registers
i,j –indexes for the active input MF for first input univers
k,l –indexes for the active input MF for second input univers
C-indexes for the active output MF

Figure 8: Simplified data path for the fuzzy controller.

All active output MFs are alpha cut and, as mentioned before, these are also stored in MF output parameter registers.

e.) In the next step the gravity centers and MF alpha cut area are processed.

f.) In the last step the controller output is processed using the center of gravity defuzzification algorithm.

E. Defuzzification

The center of gravity algorithm was used for defuzzification. For triangle type membership functions, two equations for the defuzzification algorithm are presented, one for MF gravity center estimation and another one for alpha cut area computing presented in the following equations:

$$A_i = \mu_{C_i} L_i \left(1 - \frac{\mu_{C_i}}{H}\right) \tag{7}$$

$$g_i = \frac{a_i - b_i}{6}\mu_{C_i} + \frac{a_i - b_i}{2}, \tag{8}$$

where A_i is the area, g_i is the center of gravity of i-th conclusion alpha cut membership function, μ_{C_i} -alpha cut value, H –max. value of membership function (resulted from bits used for MF value coding), $L_i = a_i - b_i$ -length of the membership function. For a general triangle based membership function with alpha cut value increase, the center of gravity (g_i) is disposed on a non-linear curve. In our algoritm, this curve is approximated with a line, whose parameters are very simply composed based on $a_i - b_i$ and μ_{C_i} (*Fig. 9.*).

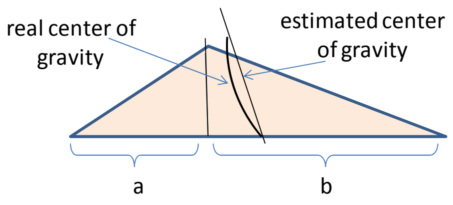

Figure 9: Estimated center of gravity for an asymmetric triangular type MF.

Using the calculated area and estimated gravity for output fuzzy sets, the controller output is processed using the following equation:

$$u = \frac{\sum_{i=1}^{N} A_i g_i}{\sum_{i=1}^{N} A_i} \tag{9}$$

The defuzzification algorithm can be simplified by using the singleton type MFs for the output space. In this case, the output MFs center and areas do not need to be computed. The MF center is known, and instead of the area, the alpha cut values are used.

F. Fuzzy controller parameters

Input spaces are coded on 12 bits. The MFs are stored in a memory with a 13-bit address space and 10 data bits, composed of 5 Block RAMs, 13 address bits and two data bits.

Out of the total number of Block RAMs, 3x5 are used for membership function storage and 1 for rule table storage. For data representation, integer based arithmetic was used. Floating point arithmetic can be used by including floating point IP CORES. The system will be tested on NEXYS-2 and VIRTEX 2PRO development boards. The FPGA implemented fuzzy controller is connected to the PC on the development board's USB port [2], [3], [9], [10]. The presented structure can also be implemented on microcontroller based systems. For data coding and MFs representation, the ideas used in neural networks for data coding and activation functions implementation can be applied [1].

5. Simulation-based measurement results

Simulations were made using the following membership functions for two input universes, represented by the robot's distance to the next target point and the robot's orientation (*Fig. 10.*) respectively the output space for the DC motors' control signals (*Fig. 11.*).

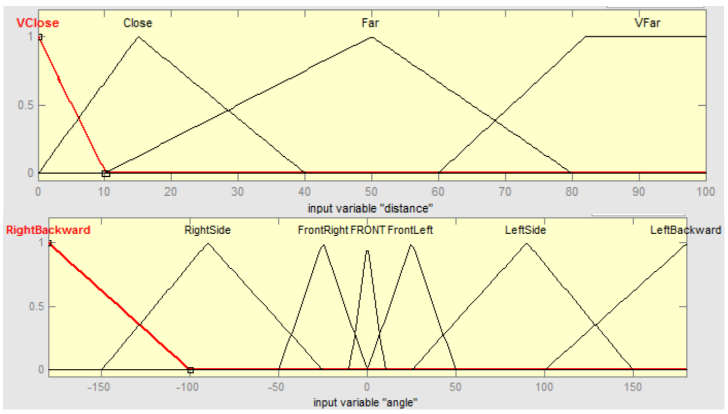

Figure 10: Membership functions for the input universe used in simulation.

The two output spaces are identical, and only the membership function of a single output space is presented (*Fig. 11.*). For the simulation experiment, the robot model described in equations (1), (2), (3) has been used.

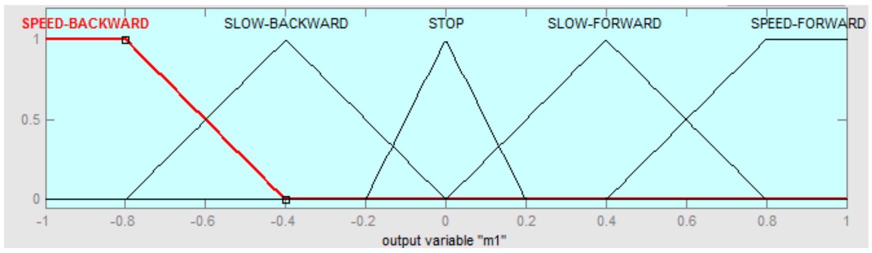

Figure 11: Membership functions for output universe used in simulation.

The membership functions for the distance input space are VClose, Close, Far and VFar, meaning that the robot is very close, close, far and very far to the target point. The second input space (robot orientation) was covered with the following MFs: RightBackward, RightSide, FrontRight, Front, Front Left, LeftSide and LeftBackward. RightBackward and LeftBackward mean that the target point is behind the robot on the right or the left part. RightSide and Left Side mean that the target point is on one side of the robot. In case of FrontRight and FrontLeft MFs, the target point is in front of the robot either on the right or on the left. The Front MFs means that the target point is directly in the front of the robot. For output MFs coding, a number of five membership functions were used (STOP, SLOW BACKWARD and SLOW FORWARD, SPEED BACKWARD and SPEED FORWARD) with a lower and a higher wheel speed control for backward and forward direction.

The robot's position for a predefined sinusoidal trajectory is presented in *Fig. 12.* for axis X and *Fig. 13.* for axis Y.

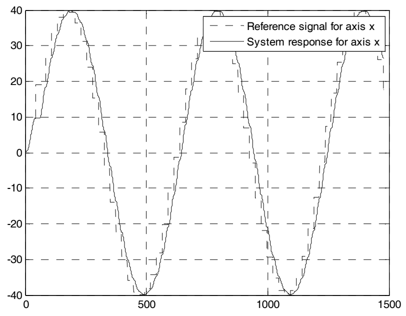

Figure 12: Reference and target trajectory for axis X.

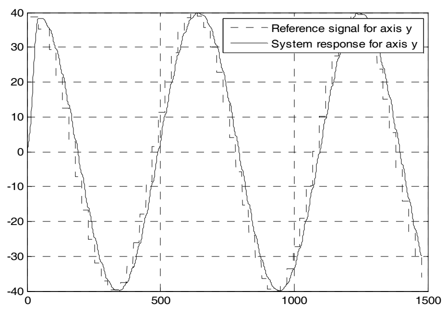

Figure 13: Reference and target trajectory for axis Y.

6. Conclusion

The hardware structure proposed for FPGA implementation presents some advantages, such as:

- Data representation on different bus sizes: in VHDL coding, the user can change/define the number of bits used for different data paths representation, such as input space coding, output space coding, membership function value coding.
- Flexible membership function programming: for all input and output spaces membership functions with different parameters can be defined.
- High speed controller output processing: the proposed mixed parallel pipeline structure permits a high order of data processing parallelism.
- Accelerated algorithm for defuzzification implementation: for triangular membership functions, the presented method reduces the defuzzification processing time.
- Easy rule base programmability: the rule base table and the membership functions can be reprogrammed from a host computer.

Acknowledgements

This paper presents a part of the achievements of a research project supported by the Institute for Research Programmes of the Sapientia University.

References

[1] Omondi, A. R., Rajapakse, J. C., FPGA Implementations of Neural Networks, *Springer*, 2006.
[2] Volnei A. Pedroni, Circuit Design with VHDL, *MIT Press, Cambridge, Massachusetts, London*, England, 2004
[3] Pong. P. Chu, FPGA Prototyping by VHDL Examples XILINX SPARTAN-3 Version, *John Whiley & Sons*, Hoboken, New Jersey, 2008
[4] Ross, J. T., Fuzzy logic with engineering applications / *Timothy J. Ross, John Wiley*, Chichester, 2004
[5] Lantos B., Fuzzy systems and genetic algorithms : lecture notes / Béla Lantos ; Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar, *Műegyetemi Kiadó*, Budapest, 2002
[6] S. T. Brassai, Neuroadaptive Systems based on FPGA circuits with application in automatic control, *Phd thesis*, Transylvania University of Brasov, Brasov, 2008
[7] http://mcu.emea.fujitsu.com/mcu_product/detail/MB90F352SPMC.htm
[8] F2MC-16LX 16-BIT MICROCONTROLLER MB90350 Series, HARDWARE MANUAL, FUJITSU SEMICONDUCTOR, CM44-10132-1E
[9] Pong. P. Chu, RTL Hardware Design Using VHDL, *John Whiley & Sons*, Hoboken, New Jersey, 2006
[10] Spartan-3 FPGA Family Data Sheet, Product Specification, XILINX, June 25, 2008