People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering
Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

MASTER

In **Electrical and Electronic Engineering**
Option: **Computer Engineering**

Title:

Fire Suppression System Based on MATLAB Video Processing & FPGA

Presented by:

- **Messi Salaheddine**
- **BOUROUIS Bilal**

Supervisor:

Dr. MAACHE Ahmed

Dedication

I dedicate this work to my dear parents who were present and supportive in the good times and the bad ones. To my family members: my sisters, my little brother, and my grandparents for their love and endless generosity.

This work is also dedicated to all my friends that I know & been involved in my life, who were at my side at all times, gave me joy and good times. To my best friends, my roommates Ayoub and Fouad, my crazy friend Nabil, and to my marsian colleague and friend Bilal.

Salaheddine

Dedication

I would like to dedicate this modest work to My dear parents, who are and will be always the most important people in my life, their care, endless support and generosity will never be forgotten.

My nice brothers and sisters, whom I hope to realize their goals in life. My friends, and all the people in my life whom I have found and been supported by, in the times of need.

Bilal

Acknowledgement

First and foremost, we must praise Allah, the Ever-Thankful, for His help and bless. This work would never be achieved without his help.

We would like to express our sincere gratitude to all the people who contributed to the fulfilment of this project especially our teacher and supervisor Dr. Ahmed Maache for his valuable support and guidance throughout all the challenges and obstacles found in the course of achieving this project.

We also are so grateful to our families, especially our parents, for their generous and endless support throughout our entire life and particularly in our education.

Last but not least, deepest thanks to all our teachers at the institute who taught us what a good engineer should know and learn, and lead us to a good ending of an unforgettable journey. Without forgetting all staff members and employees at the institute.

Abstract

Class A fires (fires out of wood, paper, cloth…etc) are the most common firebreaks in confined space, such as houses, desks, schools…etc. In this project a quick and reliable automatic fire suppression system is designed and implemented. It uses a camera to monitor the confined space at all times. Image processing techniques are used to treat the video stream and starts an alarm whenever a fire is detected. The system uses RGB components to detect the RGB characteristics of fire, and uses luminance and color information stored in YCbCr components to confirm the rules of detection. The system can locate the fire with high precision based on the geographic location of camera and location of fire pixels in the image.

After locating the fire an FPGA board is instructed to point a water cannon towards the fire source. But first Confirms the fire using a flame sensor, then an alarm is started and water is ejected from the cannon. The whole process takes matter of some seconds before action is taken. The system has proved itself to be very accurate in small monitored spaces as traced in the objectives. Further improvement have to be made before it can be installed in almost any institution where class A fires can be a risk.

Introduction

Introduction

It has been proven that humans have been able to control fire for about 1.9 million years ago. The control of fire made a huge leap in human prosperity. Making fire allowed people to cook food, to enlighten the darkness of night, keeps them warm in coldness of the winter. But it can be a devastating element when it is uncontrolled like the burning of forests, fields, domestic areas or residential areas.

Nowadays, the concentration of flammable materials and substances in small workspaces such as laboratories, inventories, stores…etc. made fire breaks to happen much more likely. Therefore many surveillance systems were designed to detect the early start of fire and start an alarm. Most of these systems rely on an excessive amount of sensors to accurately detect fires, making their implementation expensive. In addition sensors are impractical in open areas or outdoor where the sensors should be close to the source of fire.

Nowadays with the rapid development of cameras technology, high resolution digital camera became cheaper than a flame sensor. Image and video processing are living an explosive growth in the techniques and algorithms of detection of object, shapes and colors. Making computer vision imposing itself as the new trend for physical detection.

In this project a fire extinguishing system is designed using MATLAB and FPGA. In which a camera is constantly capturing frames from the area of surveillance, and transferring them to a computer running MATLAB environment. Computer vision based algorithm is developed using MATLAB script to detect fire, locate it, trigger an alarm and extinguishes. The extinction is done using a mechanical system of stepper motors as main part, controlled by FPGA board. The board controls two motors to point a water cannon straight to the fire and pump out the water from a tank.

The project attempts to provide reliable solution to fire breaks. Detection and quick action is taken in the purpose of extinguishing fire before spreading and making unrecoverable damages.

Literature review

Fire has as symptoms Flames and smoke. These two are primary signs that are used to detect fire in many computer vision algorithms.

Smoke algorithms are good in open areas, where the fire size is not so small so that it gives-off smoke, as early stage forest fires. The paper “An Early Fire Detection Algorithm Using IP Cameras” [1], presents a good algorithm to detect fires effectively, in which error with false negative and false positive error rates approximately equal to 4% and 2%, respectively.

In the proposed scheme, firstly the candidate smoke regions are estimated using motion and color smoke properties. Next using morphological operations the noise is reduced. Finally the growth properties of the candidate smoke regions are furthermore analyzed through time using the connected component labeling technique. But sometimes smoke is not sufficient for detection because its density in the air is not visible to the camera. Especially without excellent cameras supporting high resolution (HD) the efficiency of the system could well be just tolerable.

Many algorithm today combine the fire characteristics of movement and color to get better efficiency. As in the paper “A FAST AND EFFICIENT VISION BASED APPROACH FOR DETECTION OF FIRE IN REAL TIME SCENARIO”. The main consideration in this proposed technique is that, it is based on three algorithms, mainly used for flame detection: motion detection algorithm, edge detection algorithm and color detection algorithm. The results have shown that a success of 94.6% for candle fire, and 91.4% for paper fire. But the design and processing of such algorithms can be costly in time and hardware resources. Having a project with mechanical parts to take quick action and cope quickly with fire the response of the system to the changes can be a bit slow [2].

In the paper entitled “A Review on Flame and Smoke Detection Techniques in Video’s”, four techniques where used to achieve good efficiency. The first system consisting of real-time flame detector which combines statistical color information with foreground information. But this type of systems is good for fixed cameras in a static environment. So it does not suit our purpose for having an adaptive system for any environment. Second algorithm uses a lookup table for recognition of fire in video sequences using a manual training set. The lookup table is used to identify the possible fire pixels of color. This algorithm is based on artificial intelligence which is not in our scope of knowledge [3].

Third system is based on smoke detection as explained earlier.

The last proposed algorithm is a system based on two models. One based on luminance and second based on chrominance. Fuzzy logic uses the YCbCr color space for the separation of luminance from chrominance instead of using color spaces such as RGB. Existing historic rules are replaced with the Fuzzy logic to make the classification more robust and effective. This model achieves up to 99.00% correct fire detection rate with a 9.50% false alarm rate. Due to the ease of manipulating the RGB and the YCbCr color spaces and the outstanding results that are evidenced in the previous technique in the paper, we are settled on adopting the technique of extracting luminance and chrominance characteristic of flame from the YCbCr image, backed up by RGB thresholding to detect fire pixels.

Project objectives

- Construct a cheap and effective alternative, to the existing costly pre-engineered systems.
- Challenge the world's number one fire suppression system, water nozzle system. By providing a product that targets only the fire and not the whole room or building, avoiding the damage of electronic devices.

For our system to be robust, we need to ensure the following:

- Accurate detection of fires of class A, with low error rates.
- Precise location of the fire source from the video stream frames.
- Accurate positioning of the mechanical system.
- Quick response of the whole system.
- System can cope with the fire changes effectively (presence of flame, movement of flame, absence of flame).

Chapter I Theoretical Background

I.1.1 Overview on fires and extinguishing systems

Fire is hot because conversion of the weak double bond in molecular oxygen, O_2, to the stronger bonds in the combustion products carbon dioxide and water releases energy (418 kJ per 32 g of O_2). At a certain point in the combustion reaction, called the ignition point, flames are produced. The *flame* is the visible portion of the fire. Flames consist primarily of carbon dioxide, water vapor, oxygen and nitrogen. If hot enough, the gases may become ionized to produce plasma. Depending on the substances alight, and any impurities outside, the color of the flame and the fire's intensity will be different [4].

Fire has been classified into 5 classes depending on the fuel type [5]**:**

- ***Class A:*** Ordinary combustible materials, such as wood, cloth, paper, rubber and many plastics. They burn with an ember and leave an ash. Extinguish by cooling the fuel to a temperature that is below the ignition temperature. Water and other extinguishing agents are effective.
- ***Class B:*** Flammable liquids (burn at room temperature) and combustible liquids (require heat to ignite). Petroleum greases, tars, oils, oil-based paints, solvents, lacquers, alcohols, and flammable gases. High fire hazard; water may not extinguish. Extinguish by creating a barrier between the fuel and the oxygen, such as layer of foam.
- ***Class C:*** Fuels that would be **A** or **B** except that they involve energized electrical equipment. Special techniques and agents required to extinguish, most commonly carbon dioxide or dry chemical agents. Use of water is very dangerous because water conducts electricity.
- ***Class D:*** Combustible metals, such as magnesium, titanium, zirconium, sodium, lithium and potassium. Most cars contain numerous such metals. Because of extremely high flame temperatures, water can break down into hydrogen and oxygen, enhancing burning or exploding. Extinguish with special powders based on sodium chloride or other salts; also clean dry sand.
- ***Class K:*** Fires in cooking appliances that involve combustible cooking media (vegetable or animal oils and fats).

Since we want to replace the conventional nozzle system used in almost every hotel, hospital and in houses, aiming to extinguish fires of class A, we are only interested in class A fires which has orange-red color and the extinguishing agent is water.

Automatic fire suppression systems control and extinguish fires without human intervention. Examples of automatic systems include fire sprinkler system shown in figureII.2 (using water), gaseous fire (using inert gases and chemical agents) suppression, and condensed aerosol fire suppression as shown in figure II.1 (uses condensed aerosol micro-particles and effluent gases cooled and "condensed" within the device and discharged as solid particles). When fires are extinguished in the early stages, loss of life is minimal, since 93% of all fire-related deaths has progressed beyond the early stages [6]

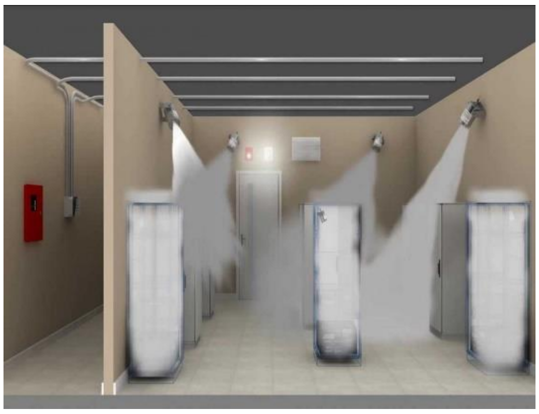


Figure I. 2 Nozzle of a mounted aerosol fire suppression system [14]

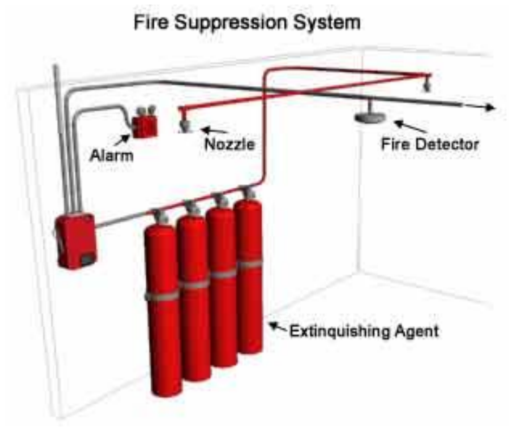


Figure I. 2 Fire sprinkler system [15]

I.1.2 Overview of Pre-Engineered Fire Suppression Systems

Pre-engineered fire suppression systems provide fast, on-site protection at the earliest stage of a fire. These systems are continuously monitoring any fire hazard in the protected area, and will be triggered automatically if any alarm conditions are met.

There are two primary types of Pre-Engineered Fire Suppression Systems. The first type is Dry Chemical Systems as in figure I.3, used for Industrial Fire Protection and utilize dry chemical compounds, Figure I.4. These chemicals suppress fire effectively and provide efficient coverage. Easy to install and maintain in any industrial setting. These systems require recharging after operation. Dry chemical agents are for the protection of ABC or BC hazards.

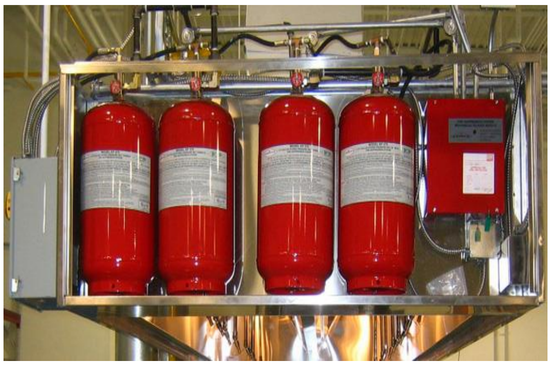


Figure I. 4 Dry chemical system. [16]

Figure I. 4 Applying Dry chemical compound. [17]

The second type of Pre-Engineered Fire Suppression System is Wet Chemical Systems, figureI.6, which are used for Commercial Cooking Area Protection. A wet agent system is an effective way for suppressing commercial cooking fires before major damage occurs. When a wet chemical agent as in figure I.5, is applied in a concentrated liquid spray to a burning surface, it reacts quickly with the cooking media (fats or oils) to produce a foam blanket covering the surface. This reaction, combined with the cooling effect of the wet chemical agent reduces the possibility of fire re-flash [7].

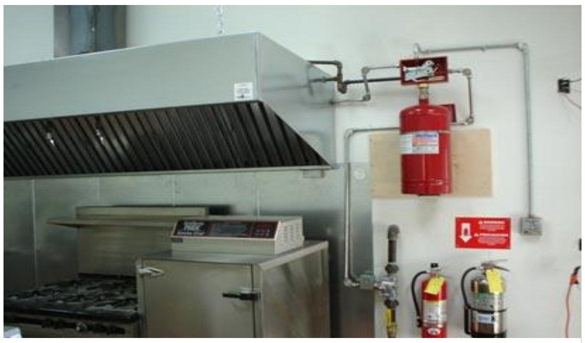


Figure I. 5 Wet chemical system. [20]

Figure I. 6 Applying wet chemical agent. [21]

Benefits of Pre-Engineered Fire Suppression Systems

- Early detection and quick response to the fire.
- Pre-engineered fire suppression systems are safe and easy to use. They do not require human intervention, except for providing manual override in case of emergency.
- Pre-engineered means pretested to ensure success.
- It eliminates the fuel source automatically. The fuel or electrical source of a fire contained within cooking equipment often continues to feed the fire after it has ignited.

I.2 MATLAB and the Image Processing Toolbox

MATLAB (Matrix Laboratory) is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation.

MATLAB is complemented by a family of application-specific solutions called Toolboxes. The Image Processing Toolbox is a collection of MATLAB functions that extend the capability of the MATLAB environment for the solution of digital image processing problems.

An image may be defined as a two-dimensional function, f(x , y), where x and y are spatial coordinates, and the amplitude of *f* at any pair of coordinates (x , y) is called the intensity or gray level of the image at that point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image [8].

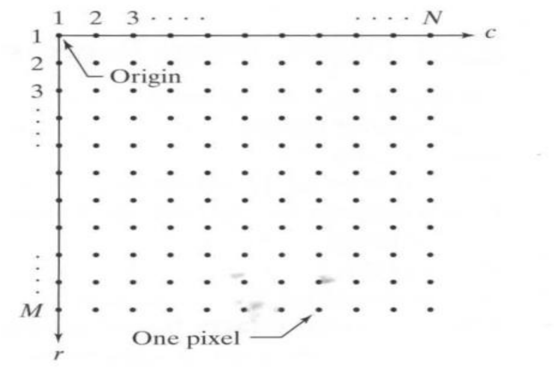


Figure I. 7 Coordinates convention in the image processing toolbox [8]

I.2.1 RGB Images:

An RGB color image is an MxNx3 array of color pixels, illustrated in figure I.8, where each color pixel is a triplet corresponding to the red, green, and blue components of an RGB image at a specific spatial location.

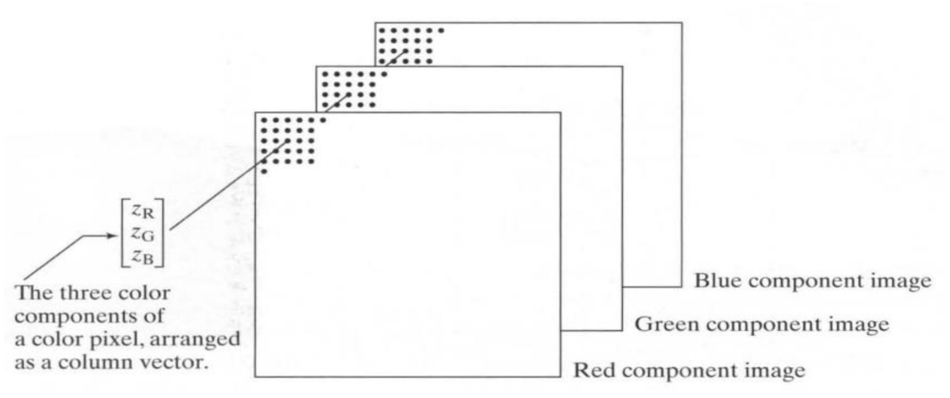


Figure I. 8 RGB image components [8]

For our fire detection algorithm for class A flames we need to apply the following rules on the red R, green G, and blue B component on each pixel of the frame acquired, but before that we need to suppress the maximum values (255) of the blue component image so that the flame will not be captured as a source of bright light:

1) Suppress the max value of blue component (255)
2) (R(x,y) > G(x,y)) AND (G(x,y) > B(x,y))
3) (R(x,y) > 190) AND (G(x,y) > 100) AND (B(x,y) < 140)

I.2.2 YCbCr Images:

The YCbCr color space is used extensively in digital video. In this format, luminance information is represented by a single component, Y, and color information is stored as two color-difference components, Cb and Cr. Component Cb is the difference between the blue component and a reference value, and component Cr is the difference between the red component and a reference value [3].

The transformation used by the toolbox to convert from RGB to YCbCr is:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \text{... (1)}$$

We need to add the following rules by using the three components of YCbCr:

4) (Y(x,y) > Cr(x,y))
5) (Cr(x,y) > Cb(x,y))

I.2.3 2-D Median Filter

Median filtering is a non-linear operation often used in image processing to reduce “salt & pepper” noise. A median filter is applied to simultaneously reduce noise and preserve edges, its effect is shown in figures I.9 and I.10.

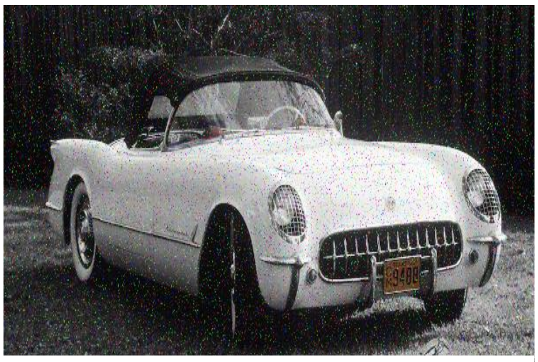


Figure I. 9 Image with salt and pepper noise [22]

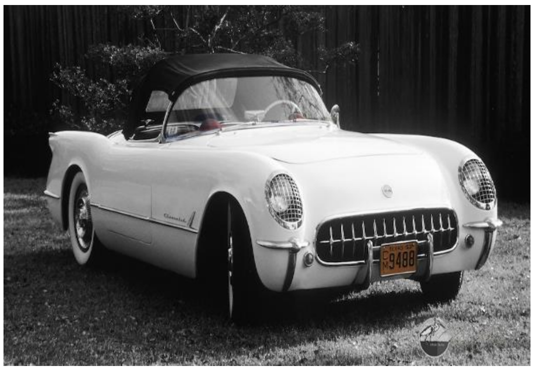


Figure I. 10 Image after applying the median filter

I.2.4 Connected Components

A binary image is produced by applying the previous rules and filter to the input image. Then compute all connected components in the binary image to locate the flames or getting the label matrix.

Let S represent a subset of pixels in an image. Two pixels p and q are said to be connected in S if there exists a path between them consisting entirely of pixels in S. For any pixel p in S, the set of pixels connected to it in S is called a connected component.

A pixel *p* at coordinates (x, y) has two horizontal and two vertical neighbors whose coordinates are (x+1, y), (x-1, y), (x, y+1), and (x, y-1). This set of neighbors of *p*, denoted N4 (*p*), is shaded in Figure I.13. The four diagonal neighbors of *p* have coordinates (x+1,y+1), (x+1,y-1), (x-1,y+1), and(x-1,y-1).Figure I.12 shows 4-connected components(N4(*p*)’) explaining these neighbors, which are denoted N4(*p*)' The union of N 4 (*p*) and N4(*p*)’ , in Figure I.11 8-connected components(N8(*p*)) are the 8-neighbors of *p*, denoted N8(*p*). Any four connected pixel is an eight connected pixel, but the reverse is not always true.

In our algorithm we are using 8-connected pixels, or an eight connected path. Toolbox function **bwlabel** computes all the connected components in a binary image. The pixels in each different connected component are assigned a unique integer , from 1 to the total number of connected components found, this is illustrated in figures I.14 and I.15 [8].

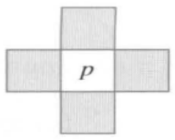


Figure I. 11 4-connected components(N4(p)) [8]

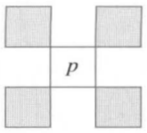


Figure I. 12 4-connected components(N4(p)’) [8]

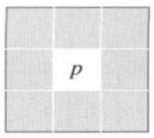
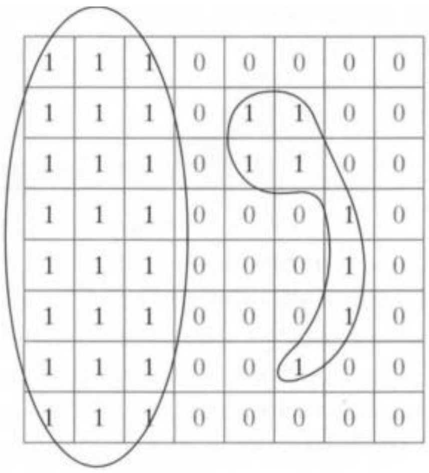
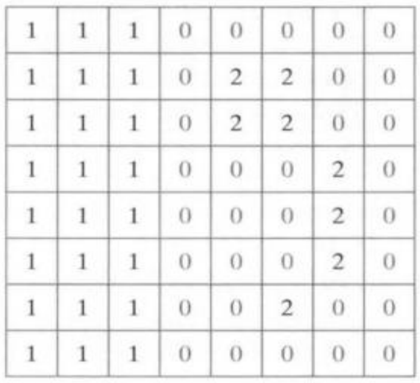


Figure I. 13 8-connected components(N8(p)) [8]



1	1	1	0	0	0	0	0
1	1	1	0	1	1	0	0
1	1	1	0	1	1	0	0
1	1	1	0	0	0	1	0
1	1	1	0	0	0	1	0
1	1	1	0	0	0	1	0
1	1	1	0	0	1	0	0
1	1	1	0	0	0	0	0

Figure I. 14 Two 8-connected components [8]



1	1	1	0	0	0	0	0
1	1	1	0	2	2	0	0
1	1	1	0	2	2	0	0
1	1	1	0	0	0	2	0
1	1	1	0	0	0	2	0
1	1	1	0	0	0	2	0
1	1	1	0	0	2	0	0
1	1	1	0	0	0	0	0

Figure I. 15 Label matrix obtained [8]

I.2.5 Regional descriptors

Function **regionprops** is the toolbox's principal tool for computing region descriptors. This function has the syntax:

D = regionprops (L, properties)

Where L is a label matrix and D is a structure. The fields of the structure denote different measurements for each region, as specified by properties. Example of properties used in our algorithm:

Area: The number of pixels in a region.

BoundingBox: 1 X 4 vector defining the smallest rectangle containing a region.

Centroid: 1 x 2 vector; the center of mass of the region. The first element of Centroid is the horizontal coordinate of the center of mass, and the second is the vertical coordinate.

This later is the property used to compute the coordinates of the center of Fire to guide the cannon towards it.

I.3 FPGA

I.3.1 Overview

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer, this configuration is than translated to a logic circuit, implemented as a hard circuit in the FPGA board.

An FPGA can be programmed to perform several logic tasks. As well as, it can be used to implement a soft microprocessor, such as Altera Nios II. The Altera DE2 board components are illustrated below in Figure I.16 [9].

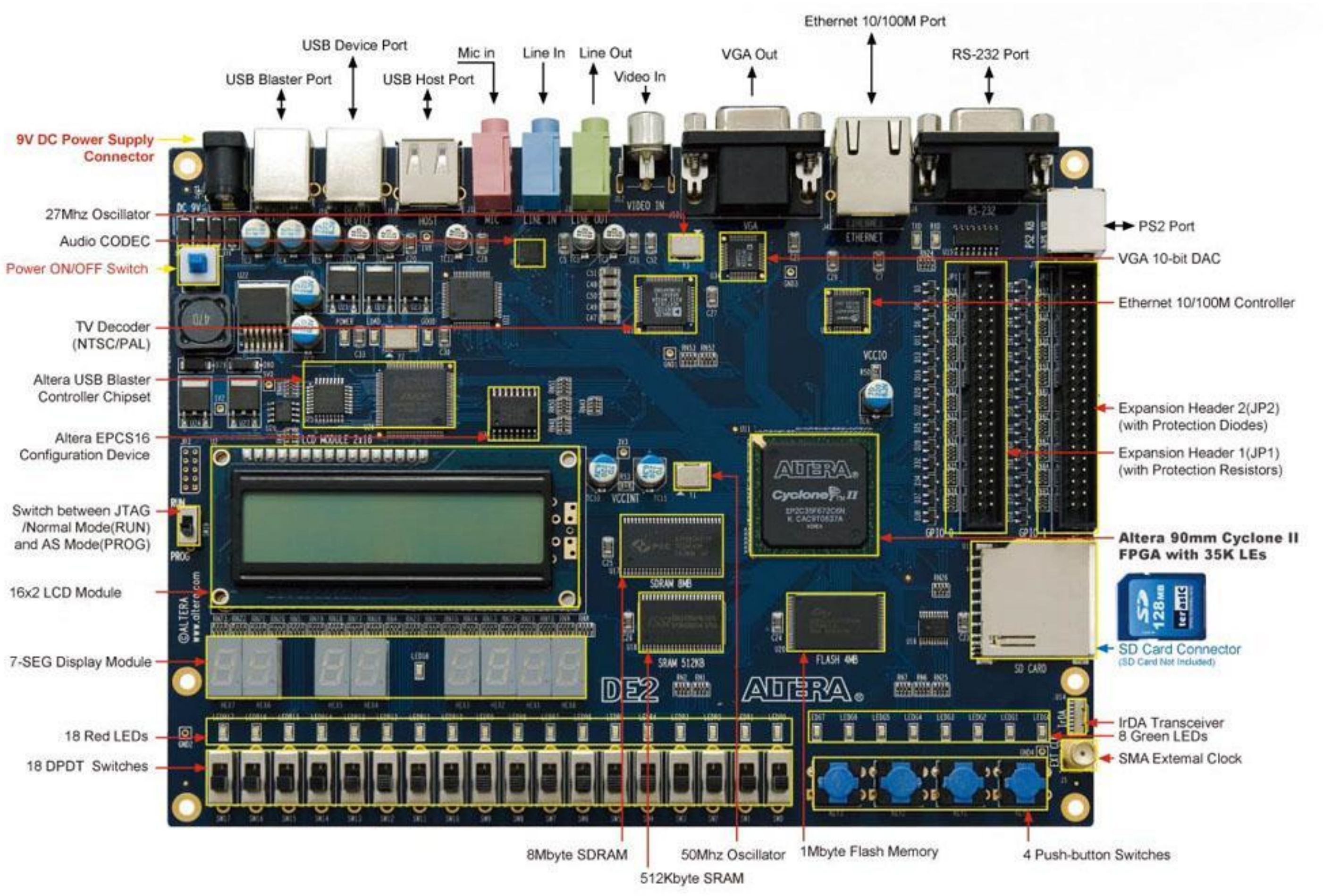


Figure I. 16 Altera DE2 Board [23]

The components of interest to this project in the DE2 board are:

- Altera Cyclone® II 2C35 FPGA device
- 512-Kbyte SRAM
- 4 toggle switches
- LEDs.
- 40-pin Expansion Headers with diode protection to connect external hardware like stepper motor drivers.

I.3.2 SOPC builder

SOPC Builder is a software made by Altera system. It is a development tool that enables you to define and generate a complete system-on-a-programmable-chip (SOPC) in much less time than using traditional, manual integration methods. It provides a powerful platform for composing memory-mapped systems based on processors, peripherals, and memories that may be internal or external to the FPGA, see Figure I.17. SOPC Builder is included as part of the Quartus II software.

The resulting "virtual" system can be connected to the outside world via the FPGA's programmable pins or connected internally to other soft components [10].

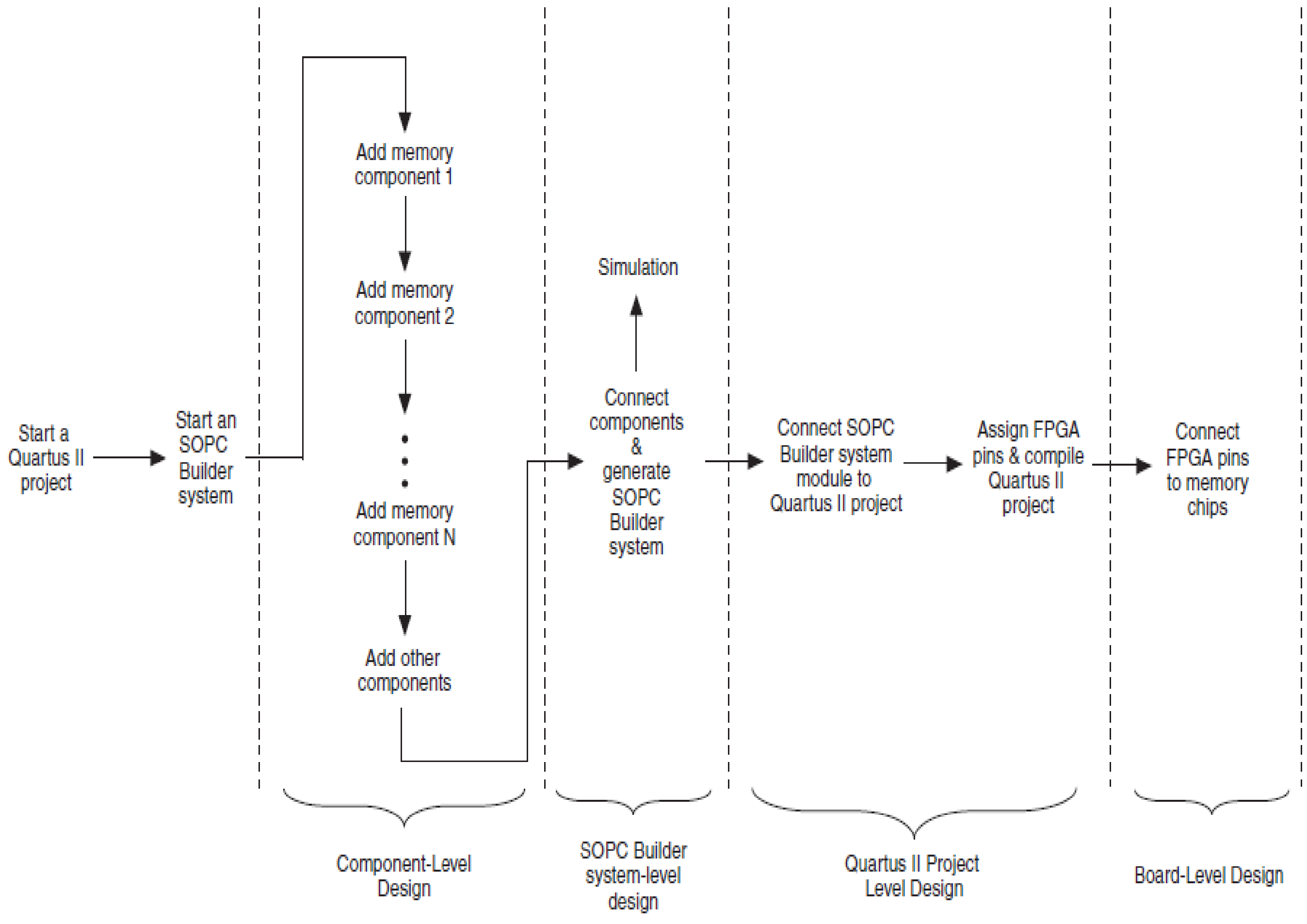

Start a Quartus II project
Start an SOPC Builder system
Add memory component 1
Add memory component 2
Add memory component N
Add other components
Simulation
Connect components & generate SOPC Builder system
Connect SOPC Builder system module to Quartus II project
Assign FPGA pins & compile Quartus II project
Connect FPGA pins to memory chips
Component-Level Design
SOPC Builder system-level design
Quartus II Project Level Design
Board-Level Design

Figure I. 17 Design flow of the SOPC builder

I.3.3 Altera NIOS II

Altera's Nios II is a soft processor, defined in a hardware description language, which can be implemented in Altera's FPGA devices by using the Quartus II CAD system. Nios II has the ability to Access a variety of on-chip peripherals. It can Interface off-chip memories and peripherals.

To implement a useful system it is necessary to add other components, as shown in figure 1.18. Such as NIOS II soft processor, 512-Kbyte SRAM memory in which the C-program can be stored, input/output interfaces that connect external hardware like stepper motor drivers, flame detector.

.

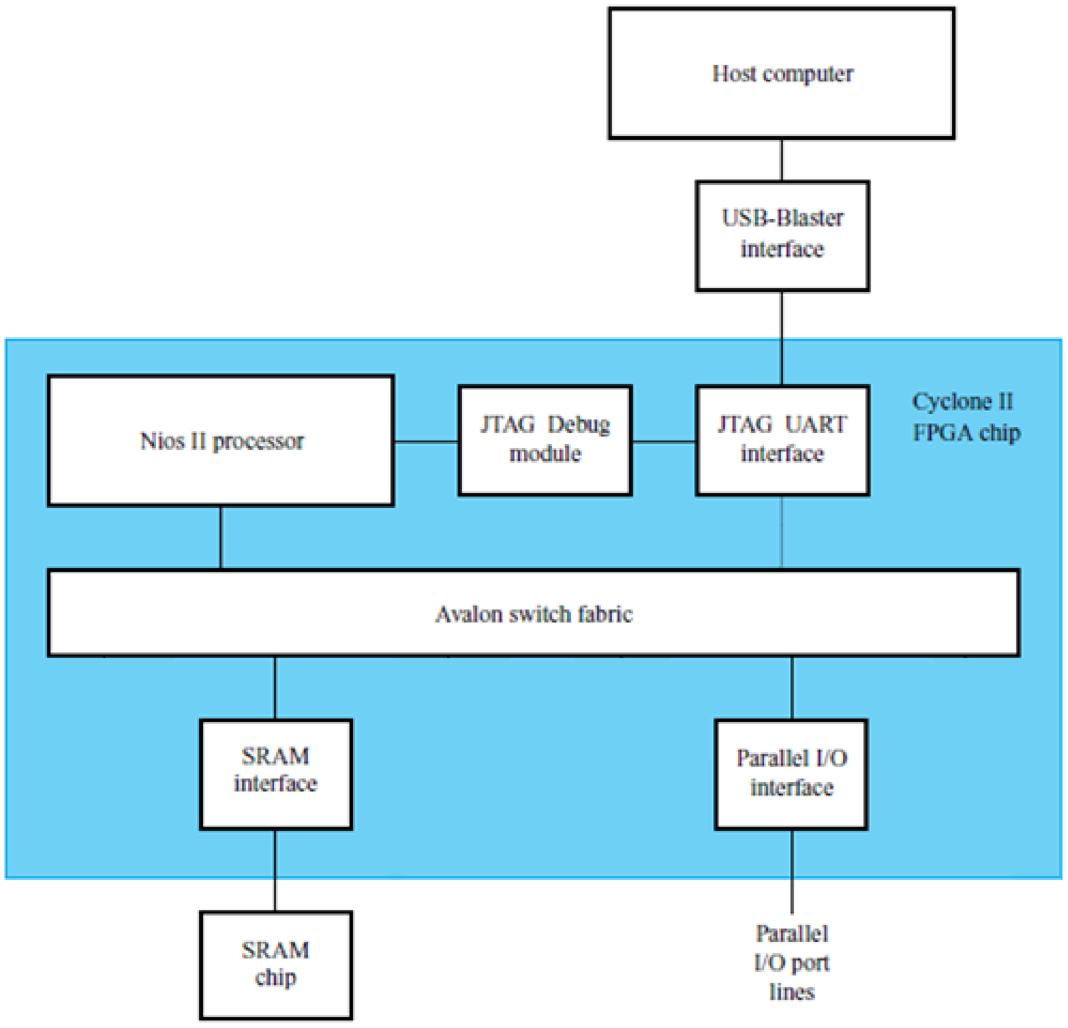


Fig II.18 System implementation using NIOS II processor

I.3.4 Nios II IDE (Integrated Development Environment)

The NIOS II integrated development environment (IDE) is the primary software development tool for the NIOS II family of embedded processors, and software development tasks can be accomplished with it, including editing, building, and debugging programs. The NIOS II IDE provides a development platform that works for all NIOS II processor systems.

After creating the SOPC system, a C/C++ application is used to run it, to execute the functions that our system is intended to do. The C/C++ application is created, debugged and downloaded into the FPGA using NiosII IDE. The Figure I.19 shows the NIOS II system design flow from SOPC to Nios II IDE to FPGA where your implementation is running.

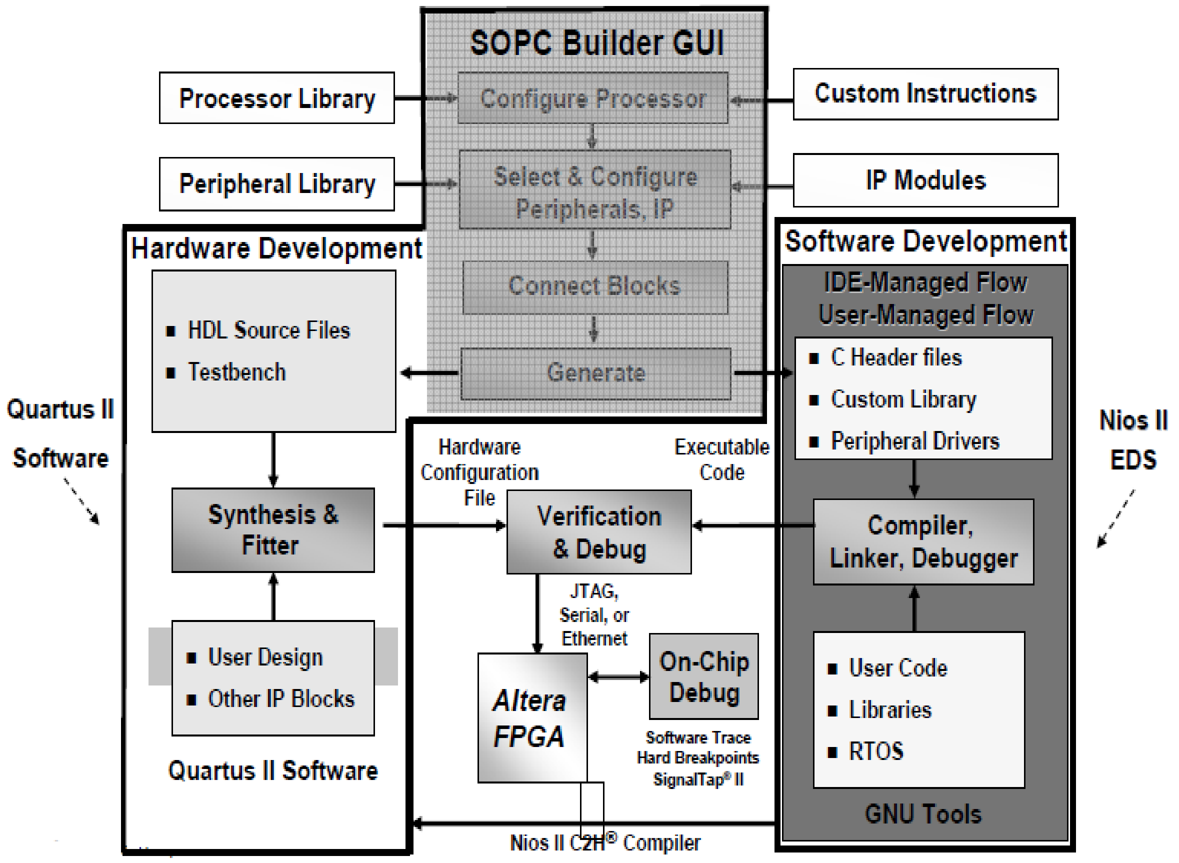


Figure I. 18 Nios II system design flow [24]

I.4 The 42BYGHW811 Bipolar Stepper Motor

I.4.1 Overview

Stepper motors are DC motors that move in discrete steps. They have multiple coils that are organized in groups called "phases". By energizing each phase in sequence, the motor will rotate, one step at a time.

With a computer controlled stepping, you can achieve very precise positioning and speed control. For this reason, stepper motors are the motor of choice for many precision motion control applications, as the precision we are looking for in our system [12]. The 42BYGHW811 bipolar stepper motor module is shown below in Figure I.20.

.

Figure I. 19 1.8° bipolar stepper motor

Advantages of the Stepper Motor:

- The rotation angle of the motor is proportional to the input pulse.
- The motor has full torque at standstill (if the windings are energized)
- The motors response to digital input pulses provides open-loop control, making the motor simpler and less costly to control.

Disadvantages of the Stepper Motor:

- Resonances can occur if not properly controlled.
- Not easy to operate at extremely high speeds.

I.4.2 The stepper motor configuration

Stepper motors can be controlled either in unipolar or bipolar modes. In bipolar mode higher efficiency can be achieved, although in unipolar mode the connection is simpler.

Unipolar:

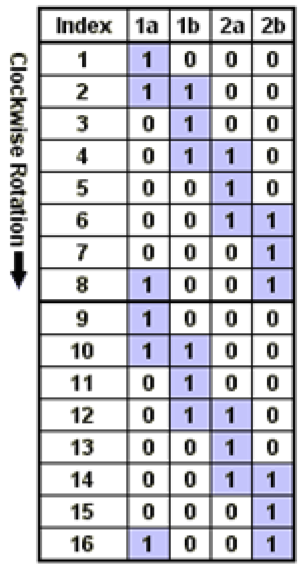
Each stator coil features a center tap with a fixed connection to the supply voltage. The current flows through the phase windings in only one direction, since each phase is wound in parallel with two wires. The current direction of the stator coil depends on which coil end is connected to earth. This configuration results in savings in the control electronics.

Bipolar

In bipolar mode each phase windings of the motor is supplied via a full bridge and can therefore carry current in both directions. Therefore the polarity of the whole coil is reversed in all instances, and not just one half. If the stepper motor phases are wound with two parallel wires, the two branches of a coil have to be connected in parallel.

Control sequences

Figure I.21 below shows the half step sequences that are used to drive the stepper motors:


Clockwise Rotation

Index	1a	1b	2a	2b
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1
9	1	0	0	0
10	1	1	0	0
11	0	1	0	0
12	0	1	1	0
13	0	0	1	0
14	0	0	1	1
15	0	0	0	1
16	1	0	0	1

Figure I. 20 The half step control sequence of stepper motor

I.4.3 74HC244/74LS244 Octal tri-state buffers

Since the FPGA is operating in CMOS technology, a tri-state buffer is required for compatibility with TTL of the 74LS244 buffer, H-bridge motor driver, and the ADC. The 74HC244 octal tri-state buffer provides this interfacing compatibility.

I.4.4 L293D Stepper motor driver

As it is shown in the Figure I.20, The L293D device is designed to provide bidirectional drive currents of up to 600 mA at voltages from 4.5 V to 36 V. The L293D is also designed to drive inductive loads such as relays, solenoids, DC and bipolar stepping motors other high current/high-voltage loads in positive- supply applications.

The input pins of this driver are compatibly connected with LSTTL output coming from the 74LS244 octal buffer.

Features:

- Wide Supply-Voltage Range: 4.5 V to 36 V
- Output Current 1 A Per Channel

Application:

- Stepper Motor Drivers
- DC Motor Drivers
- Latching Relay Drivers

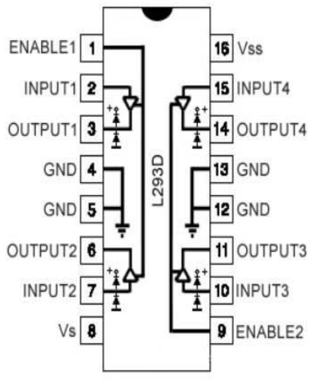


Figure I. 21 The L293D dual H bridge driver

I.5 Sensors

I.5.1 Flame Sensor LM393

The module is mainly used to detect the infrared light. It outputs digital signal 0 and 1 through a Comparator output. The output value will be 0 when infrared light is detected, otherwise it is '1'.. The sensitivity is adjustable by the precision potentiometer [13].

The Figure I.23 shows the flame sensor module used in our system:

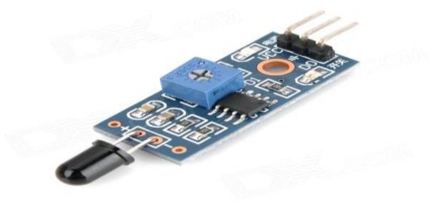


Figure I. 22 IR sensor LM393

Module specifications

- Output format: digital outputs (0 and 1).
- Maximum detection distance of 1m.
- Detection range of the infra-red wavelength: 760nm~1100nm.
- The detection angle of around 60 degrees.
- Adjustable sensitivity (shown in blue digital potentiometer adjustment in Figure I.23).
- The flame sensor is most sensitive to flame, and is generally used in flame alarms etc.

The usage of the module

The module is mainly used to detect the infrared light. It outputs digital signal 0 and 1 through a Comparator output. The output value will be 0 when infrared light is detected, otherwise its working voltage is: 3.3V-5V. The sensitivity is adjustable by the precision potentiometer.

I.5.2 The water Level Sensor

The water level sensor converts water size into an analog signal. This output analog is converted to a digital value using an ADC, the digital value can be used in a program to achieve the function of water level sensor. The module is shown below in Figure I.24:

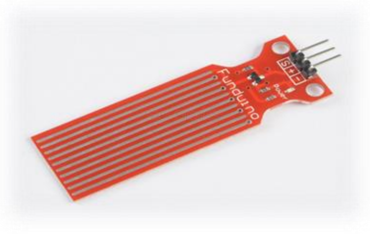


Figure I. 23 Water level sensor

I.3.3 0808ADC

The Analog to Digital converter: is an integrated circuit used for converting analog signals to discrete digital ones. The ADC outputs meet TTL level specifications, so 2 buffers are needed to interface with FPGA. The 0808 ADC has a resolution of 8 bits, shown in figure I.25 bellow.

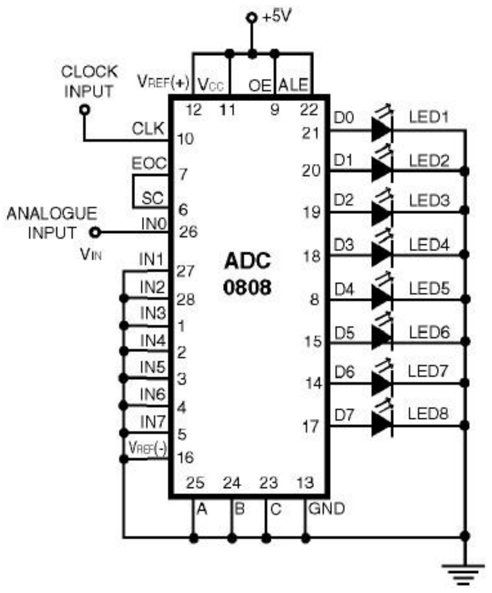


Figure I. 24 0808 ADC

Chapter II System Design

As briefed earlier in the abstract, our system is designed to replace nozzle fire extinguishers, to produce a design that is accurate and reliable.

We imagined our system to be placed in the center of a roof of an average sized room, such as a desk or a laboratory, where the camera and the reach of the cannon can cover any corner, see figure II.1.

Figure II. 1 Camera on the center.

To illustrate our design a prototype system is mounted as shown in figure II.2, the camera is above the surface of test, capturing live video stream and sending it to a Computer running MATLAB. Then, each frame of the video is processed. If a fire is detected, its coordinates are computed and sent to the FPGA and fire is confirmed using IR sensor. Finally, the FPGA guides the cannon to the indicated coordinates by rotating two stepper motors and starts the extinction process.

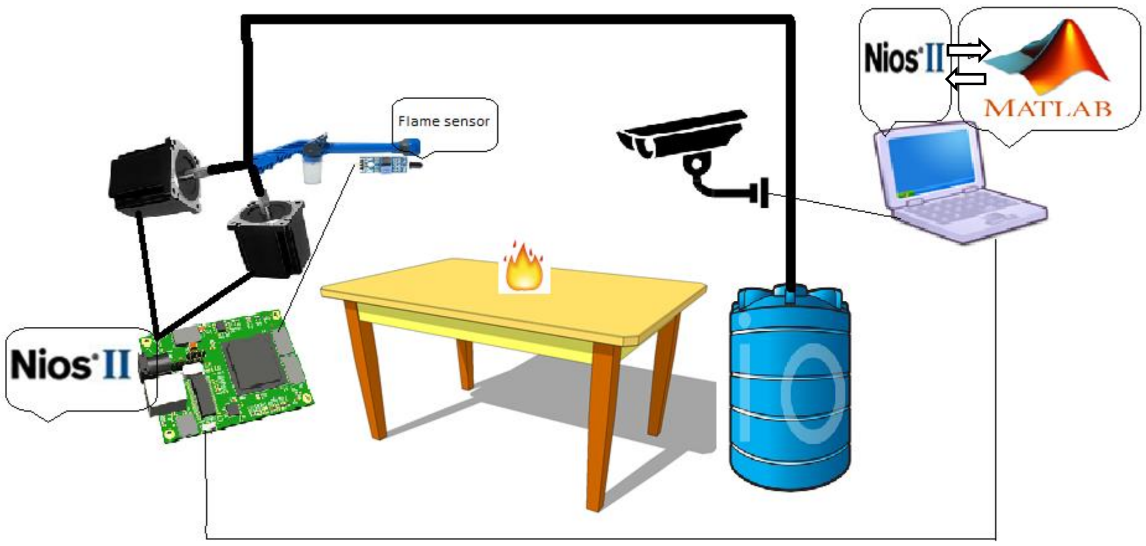


Figure II. 2 Fire extinguishing prototype system design

II.1 Software Design:

Our Software system is composed of two parts, the MATALB image processing algorithm, and the NIOS II system guiding the Cannon and extinguishing the fire.

II.1.1 MATALB image processing algorithm:

The image processing algorithm operates as follows:

Acquiring frames in RGB & YCbCr color space

First MATALB has to acquire frames from the video input coming from the webcam. Since the frames are in RGB color space we need to convert it into another variable to YCbCr color space. Next we suppress the peak value of the blue component (B=255) from the RGB frames and apply the rules stated in the theoretical part (sections I.2.1 and I.2.2). This will result in a 2-D binary image, representing the presence/absence of flames.

Enhancing the binary image

A 2-D median filter is applied on the 2-D binary image binary image to filter out the tiny connected components maybe present in the binary frame. Then, holes are filled in the regions of connected components for a better detection.

Getting image properties

We generate the label matrix using the MATLAB toolbox function: **bwlabel**, which generates an image with a set of connected components that enables us to get the required properties. These properties include the contour of the regions of fire and the center of the mean of those regions. MATLAB toolbox function: **regionprops**, extract those properties and outputs a set of coordinates of each property.

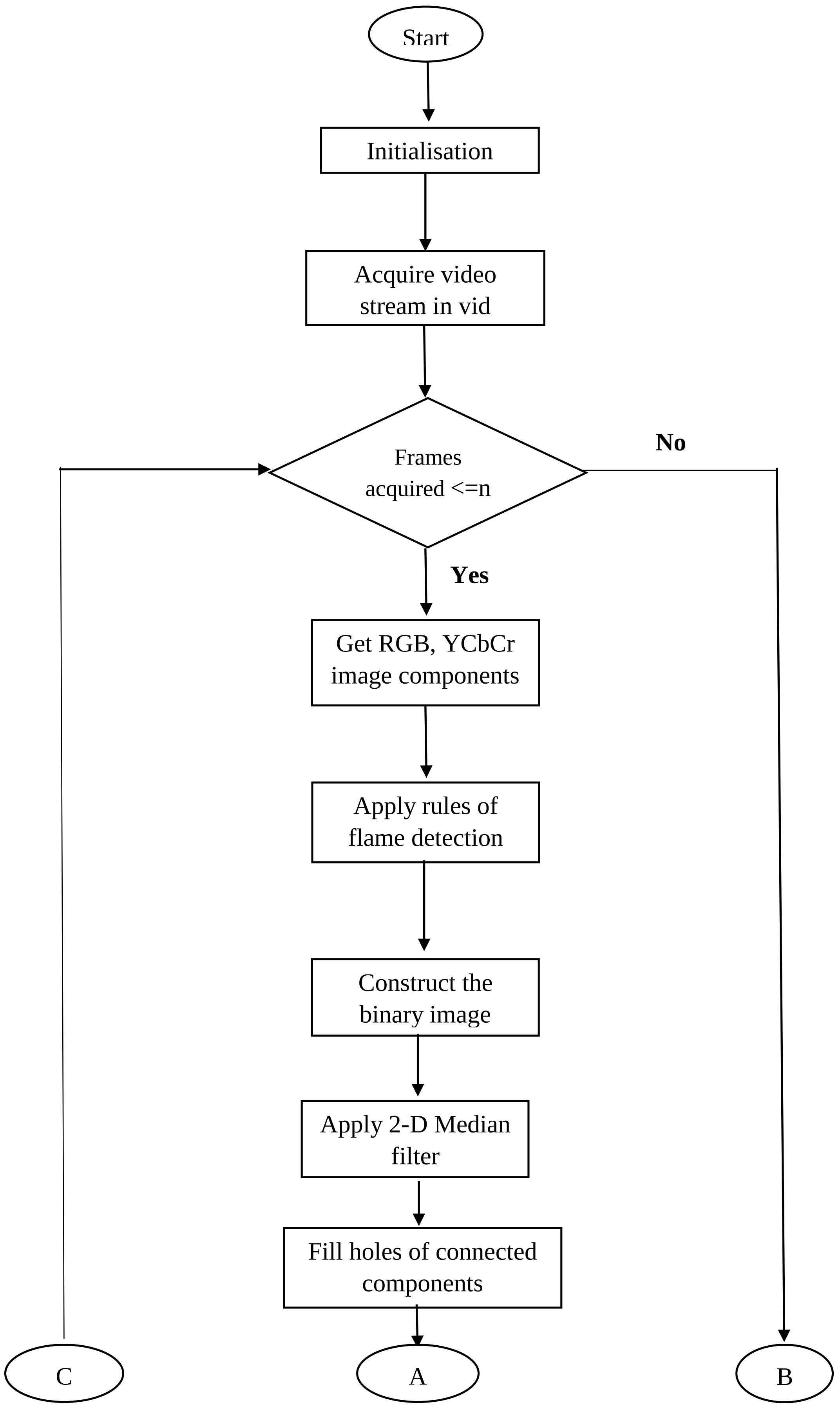
The properties we used are: the **BoundingBox** to get the contour of the flame, where we used this property to draw a rectangle around the detected flame. The centroid property is used to mark graphically the center of fire, but more importantly to translate these set of coordinates on the image to the real physical flame angles (vertical and horizontal) by which we should rotate the stepper motors to point the cannon exactly at the target flame.

Computing angles

Translating these coordinates to actual physical coordinates is performed next. A function: **angle** is written using Matlab to compute physical coordinates taking into account the distance between the camera and the motors by applying geometrical rules. This function tracks the current position of the motors and their previous position to compute the next move of tracking the fire.

Sending data to FPGA

Finally after getting the coordinates we need to pass them to the FPGA. Two methods are considered. First method is reading and writing a mutual file between the NIOS II system and MATLAB. Synchronizing is done between them using the system time. Data is send using the USB blaster. Second method consists of using an RS232 cable and protocol. But an emergency system has to be installed, in case communication fails. For example due to synchronization overlapping. A keyboard is used to enter manually the angles to be send to FPGA.


Start
Initialisation
Acquire video stream in vid
Frames acquired <=n
No
Yes
Get RGB, YCbCr image components
Apply rules of flame detection
Construct the binary image
Apply 2-D Median filter
Fill holes of connected components
C
A
B

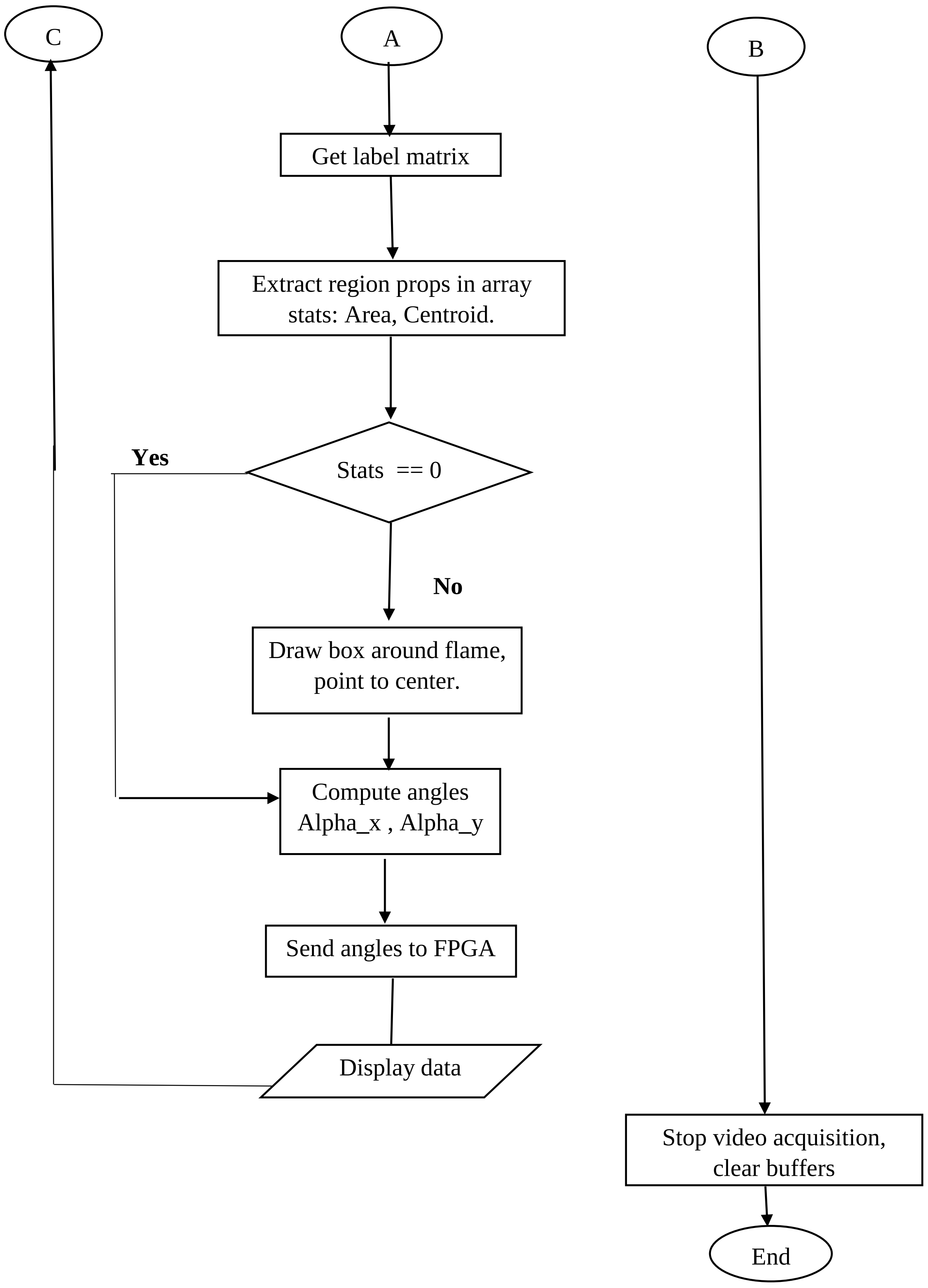

C
A
B
Get label matrix
Extract region props in array stats: Area, Centroid.
Stats == 0
Yes
No
Draw box around flame, point to center.
Compute angles Alpha_x , Alpha_y
Send angles to FPGA
Display data
Stop video acquisition, clear buffers
End

Figure II. 3 MATLAB program flowchart

II.1.2 Nios II Software Design

Checking the water level of the pump

First of all, the state of the tank will be checked based on the received data of water level. When the tank gets low on extinguishing agent (water), an alarm will be enabled to indicate that the tank has to be filled up.

Data of angle is read in order to rotate each stepper motor into a specific position.

Checking the availability of data to be read

A condition variable was introduced to organize data reception from Matlab. Based on the value of new data that is read. In order to specify an angle and a direction for the motors to rotate.

Based on received data from Matlab we calculate the required steps to move from actual position to new position in straight forward way.

$$\text{Actual position:}\, PresentStep = (int)\frac{angle}{stepAngle} \qquad \dots (2)$$

Enabling water canon & Alarm:

A water cannon starts to shoot water from tank when the flame detector has confirmed the presence of fire source. The warning alarm stays ON as long as fire is not extinguished.

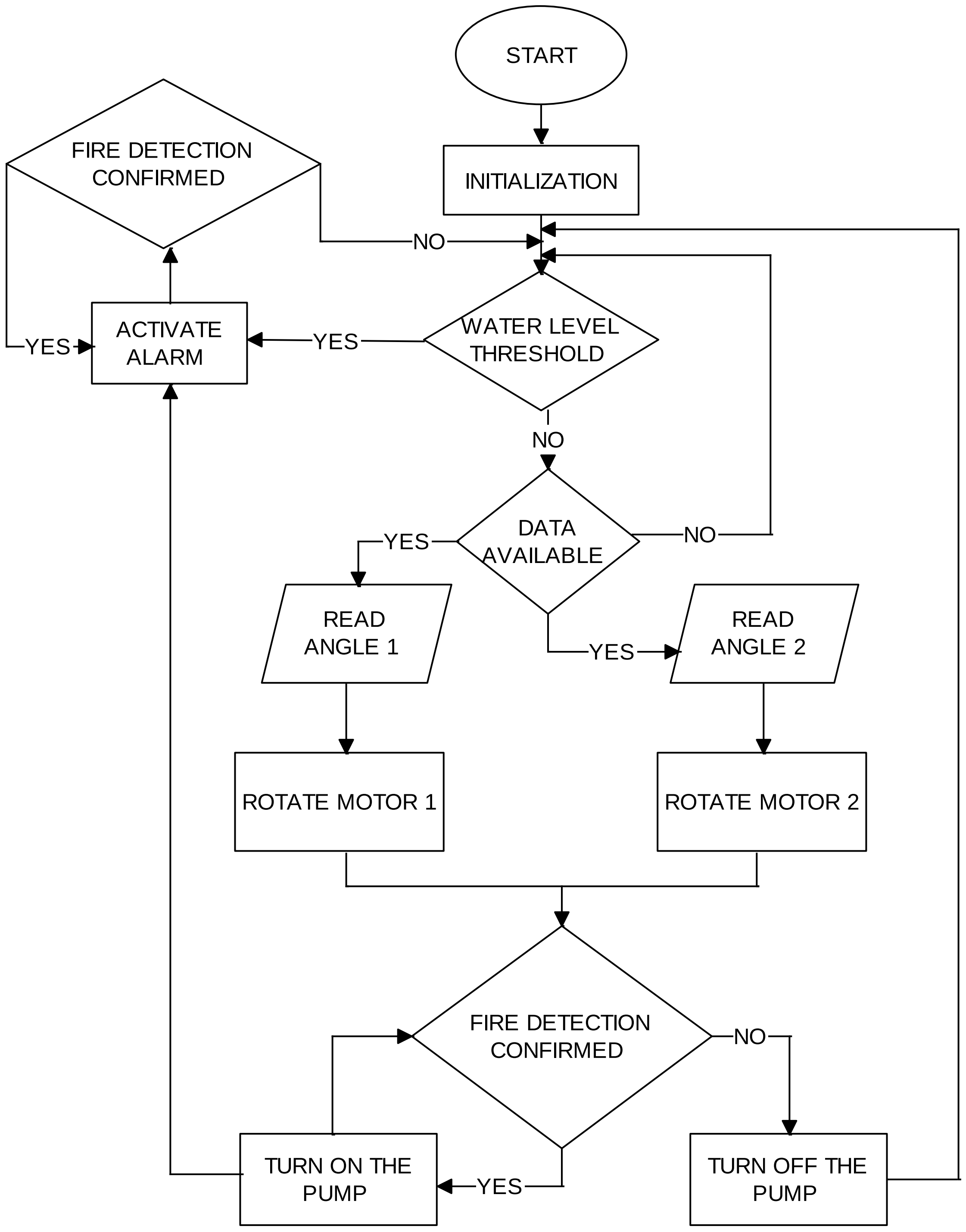


Figure II. 4 Nios II flow chart

II.2 Hardware design

The overall hardware design of the system is illustrated in Figure II.5. Shows all the main components.

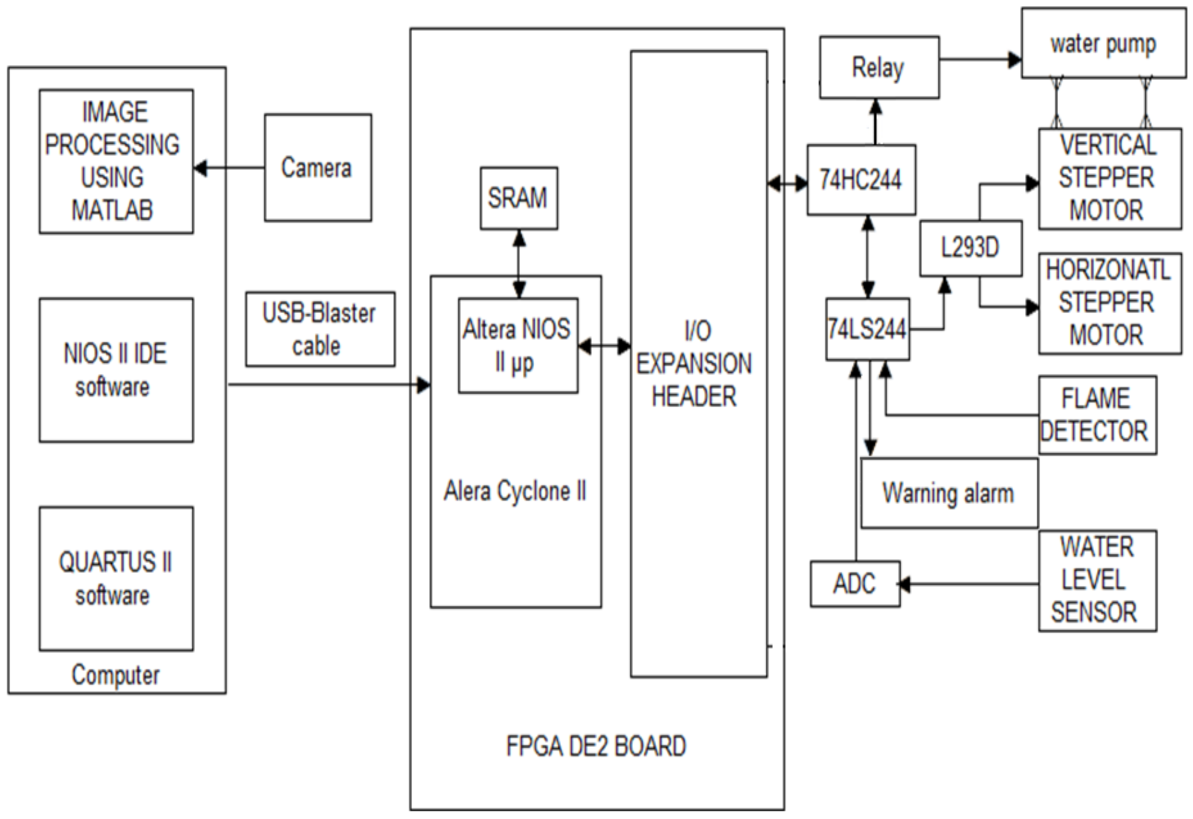


Figure II. 5 Overall NIOS II hardware design

The system works as follows:

1- The NIOS II hardware design is implemented and generated using SOPC builder tool of Quartus II software.

2- Programmable pins are assigned, system is compiled and downloaded onto the Altera Cyclone II chip.

3- A C-application will run the required functions and operations of system. Water cannon will be driven by the stepper motors in order to shoot water towards the fire source.

4- The program code will be build, debugged and loaded onto the SRAM memory of DE2 board.

5- When system starts, data are read from the ADC connected to water level sensor, in order to check the level of tank.

6- A warning alarm will be latched in case of water level threshold.

7- MATLAB software is running a video processing Algorithm in order to detect any fire presence, using a camera.

8- MATLAB will compute the corresponding angles for stepper motors to rotate, and send them to the DE2 board.

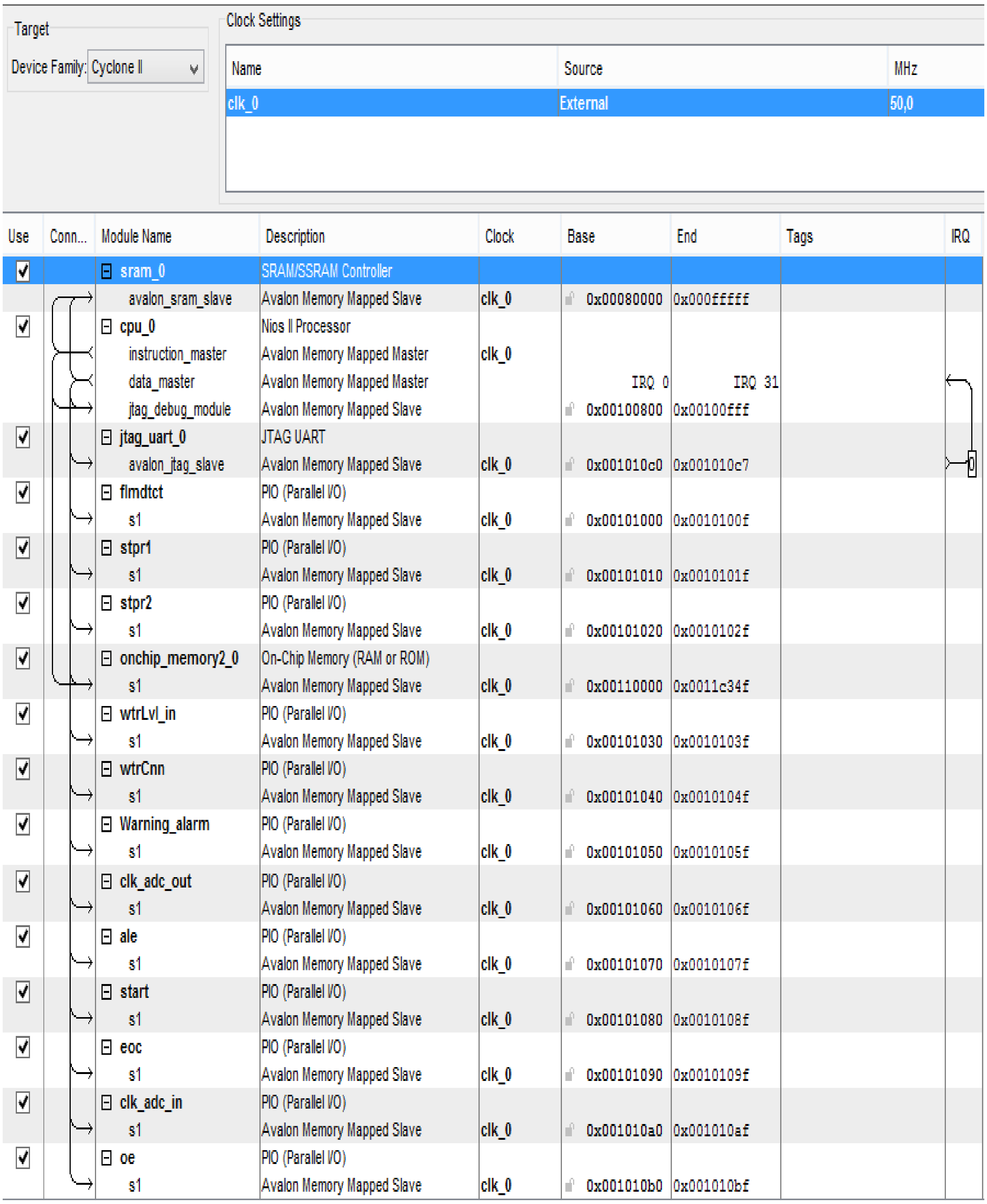
9- If the tank is filled enough, both stepper motors will be driven by the L293D H-Bridge and rotate towards the corresponding flame location.

10- If the flame sensor detects any fire radiation: water pump is started, activating the warning alarm and shooting water.

11- Otherwise, the whole system will be re-initialized, disabling all components, and driving-back both stepper motors to their reference positions.

II.3 SOPC builder Design

The SOPC hardware system design is illustrated below in Figure II.6.



Target

Device Family: Cyclone II

Clock Settings

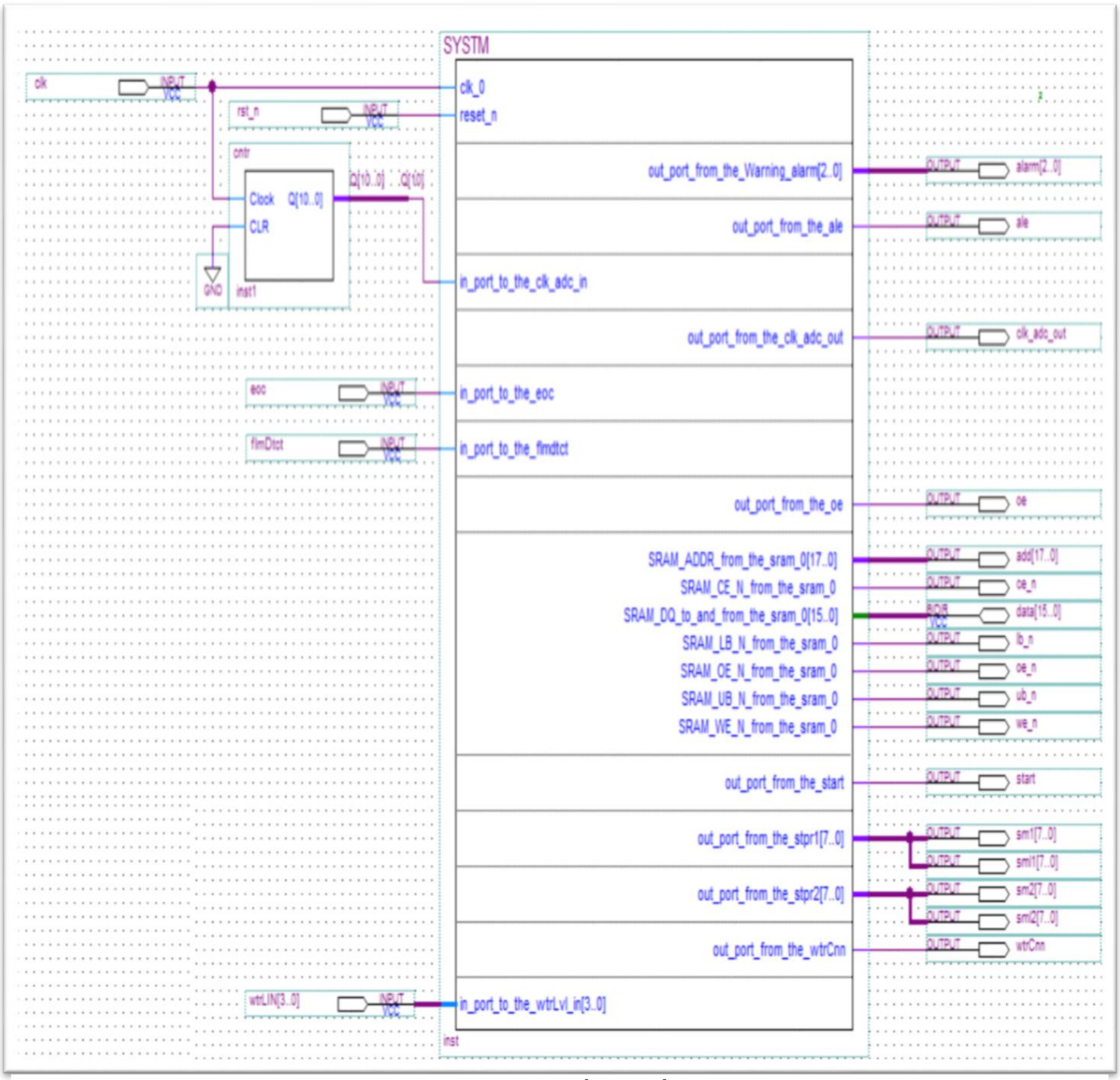
Name	Source	MHz
clk_0	External	50,0

Use	Conn...	Module Name	Description	Clock	Base	End	Tags	IRQ
✓		sram_0	SRAM/SSRAM Controller					
		avalon_sram_slave	Avalon Memory Mapped Slave	clk_0	0x00080000	0x000fffff		
✓		cpu_0	Nios II Processor					
		instruction_master	Avalon Memory Mapped Master	clk_0				
		data_master	Avalon Memory Mapped Master		IRQ 0	IRQ 31		
		jtag_debug_module	Avalon Memory Mapped Slave		0x00100800	0x00100fff		
✓		jtag_uart_0	JTAG UART					
		avalon_jtag_slave	Avalon Memory Mapped Slave	clk_0	0x001010c0	0x001010c7		0
✓		flmdtct	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101000	0x0010100f		
✓		stpr1	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101010	0x0010101f		
✓		stpr2	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101020	0x0010102f		
✓		onchip_memory2_0	On-Chip Memory (RAM or ROM)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00110000	0x0011c34f		
✓		wtrLvl_in	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101030	0x0010103f		
✓		wtrCnn	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101040	0x0010104f		
✓		Warning_alarm	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101050	0x0010105f		
✓		clk_adc_out	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101060	0x0010106f		
✓		ale	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101070	0x0010107f		
✓		start	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101080	0x0010108f		
✓		eoc	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00101090	0x0010109f		
✓		clk_adc_in	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x001010a0	0x001010af		
✓		oe	PIO (Parallel I/O)					
		s1	Avalon Memory Mapped Slave	clk_0	0x001010b0	0x001010bf		

Figure II. 6 SOPC hardware system design

II.4 Quartus Hardware System Design

The Quartus Hardware System design is shown below in Figure II.7.

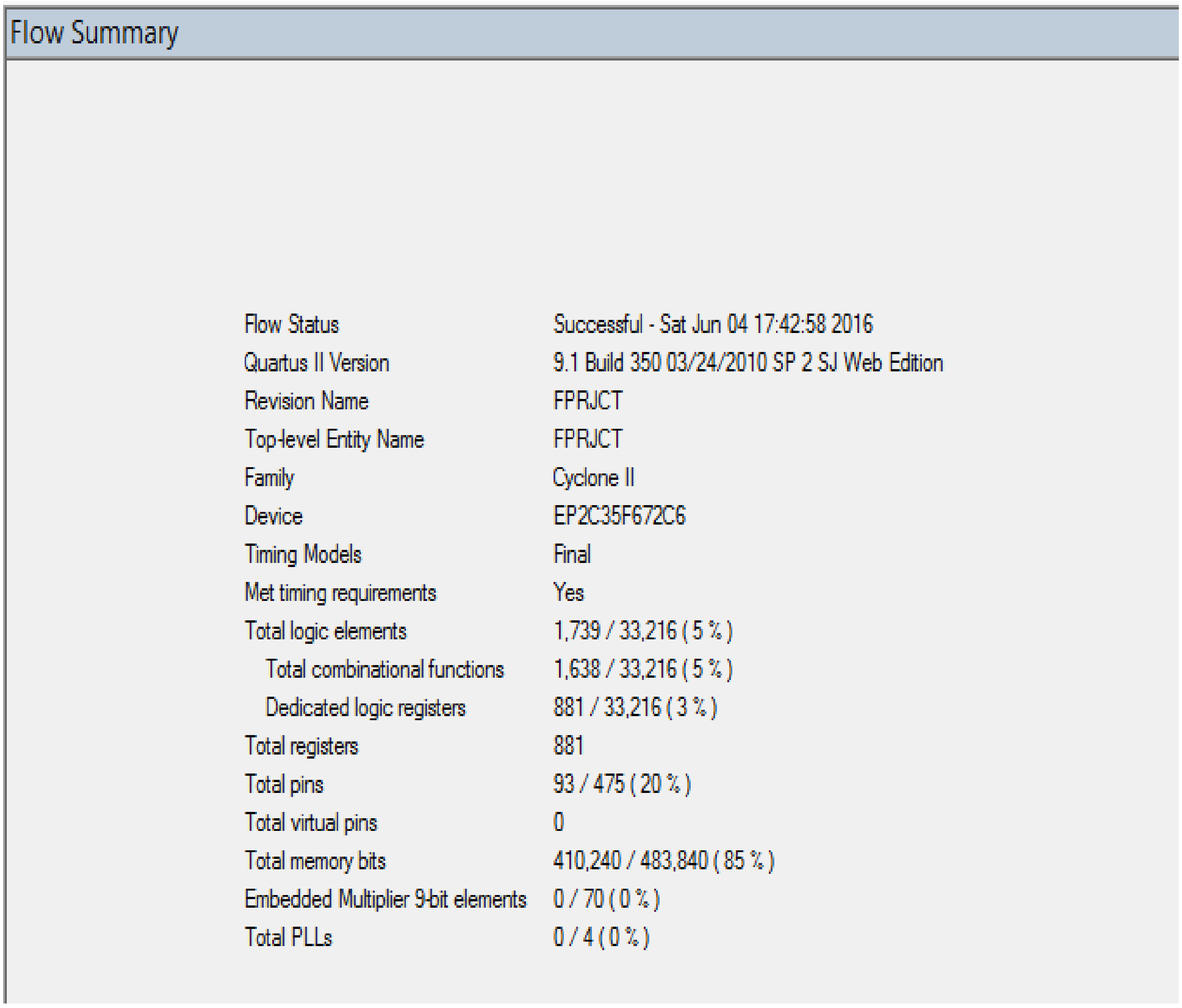


SYSTM
clk
rst_n
cntr
Clock
CLR
Q[10..0]
Q[10..0] ... Q[10]
GND
inst1
eoc
flmDtct
wtrLIN[3..0]
clk_0
reset_n
out_port_from_the_Warning_alarm[2..0]
out_port_from_the_ale
in_port_to_the_clk_adc_in
out_port_from_the_clk_adc_out
in_port_to_the_eoc
in_port_to_the_flmdtct
out_port_from_the_oe
SRAM_ADDR_from_the_sram_0[17..0]
SRAM_CE_N_from_the_sram_0
SRAM_DQ_to_and_from_the_sram_0[15..0]
SRAM_LB_N_from_the_sram_0
SRAM_OE_N_from_the_sram_0
SRAM_UB_N_from_the_sram_0
SRAM_WE_N_from_the_sram_0
out_port_from_the_start
out_port_from_the_stpr1[7..0]
out_port_from_the_stpr2[7..0]
out_port_from_the_wtrCnn
in_port_to_the_wtrLvl_in[3..0]
inst
alarm[2..0]
ale
clk_adc_out
oe
add[17..0]
ce_n
data[15..0]
lb_n
oe_n
ub_n
we_n
start
sm1[7..0]
sm1[7..0]
sm2[7..0]
sm2[7..0]
wtrCnn

Figure II. 7 SOPC schematic design

Compilation summary of Quartus II

Figure II.8 represents the resulting compilation summary of Quartus II software.


Flow Summary

Flow Status	Successful - Sat Jun 04 17:42:58 2016
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	FPRJCT
Top-level Entity Name	FPRJCT
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	1,739 / 33,216 (5 %)
Total combinational functions	1,638 / 33,216 (5 %)
Dedicated logic registers	881 / 33,216 (3 %)
Total registers	881
Total pins	93 / 475 (20 %)
Total virtual pins	0
Total memory bits	410,240 / 483,840 (85 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

Figure II. 8 Compilation summary of Quartus II

Chapter III Implementation & Results

This chapter presents the implementation of the design previously proposed, shows the results of the implemented system and highlights it's the success and limitations.

The Matlab flame detection algorithm was implemented first as it is the first stage of flame detection. The hardware circuit was then implemented to translate the computational results from Matlab into stepper motor rotations and water pump activation.

III.1 Matlab algorithm Testing & discussion

III.1.1 Testing main parts of the algorithm

Getting the binary image:

First, acquire video frames are acquired from USB webcam connected to PC. The acquired images are in RGB. The images were then converted into YCbCr components on which the rules of fire detection can be applied as stated in the theoretical part (sections I.2.1 & I.2.2) in order to get the desired binary image.

In the figures III.1 and III.2 the resulting binary image is obtained after application of the rules on the candle image, where fire from the candle is a Class A fire (orange-red color).

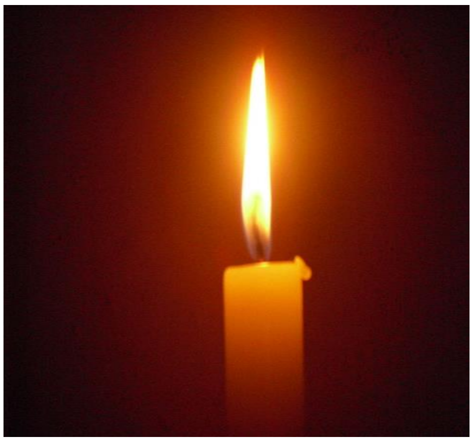


Figure III. 1 simple fire

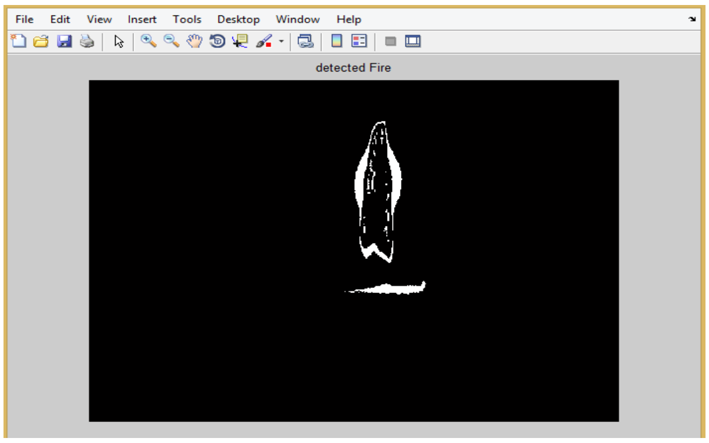


Figure III. 2 Resulting binary image.

Noise suppression

Trying the algorithm on different situations, sometimes we get a ‘salt&pepper’ noise effect in which some detected regions will appear as white dots, and they are taken into account in the region properties of the frame later in the program leading to some incorrect results about the center of fire.

Without the filter the noise can be interpreted as source of fire and trigger false alarm, although there is no fire in the scene, see figure III.3 and III.4.

Noise is suppress using a median filter, which leaves the frame clean from impurities. The following figures is an instance application of a 2-D median Filter.

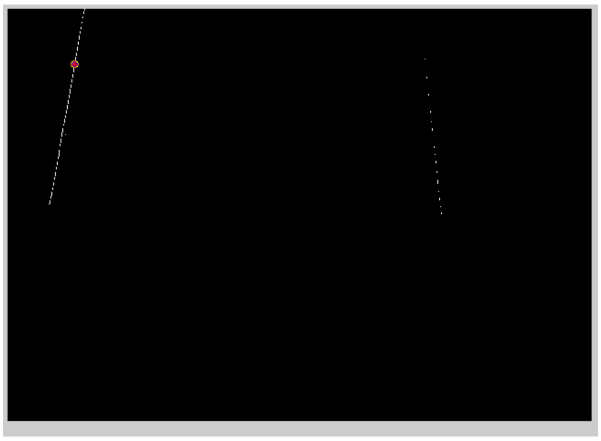


Figure III. 3 Noise interpreted as fire source where no fire is present

Figure III. 4 Applying filter, the noise is suppressed

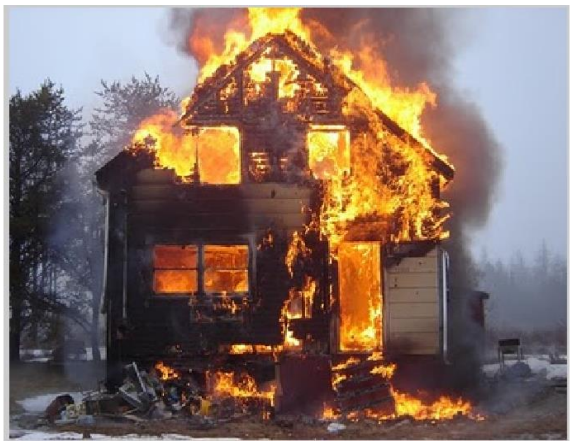


Figure III. 5 Burning house

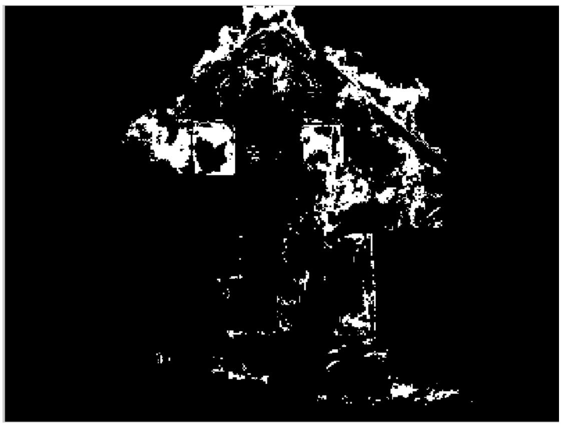


Figure III. 6 Binary image

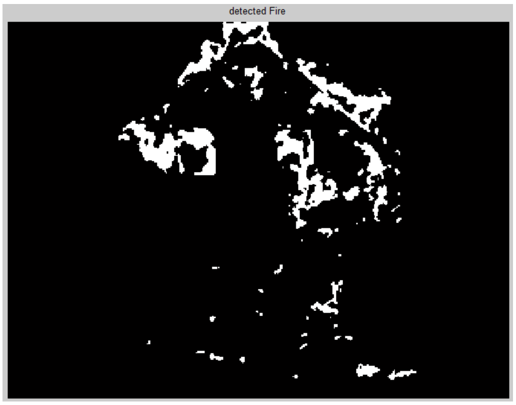


Figure III. 7Applying the 2-D median filter on the binary image

Image filling

Next for better results when computing the center of connected components in the image (center of fire), we fill the holes in the existing connected components, obtained from the median filter.

To clearly see the effect of this filling function we apply it on the previous binary image of the candle without applying the median filter.as shown below in figures III.8 and III.9.

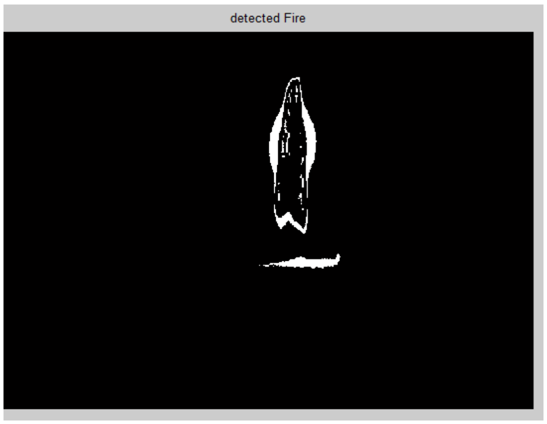


Figure III. 8 Candle binary image.

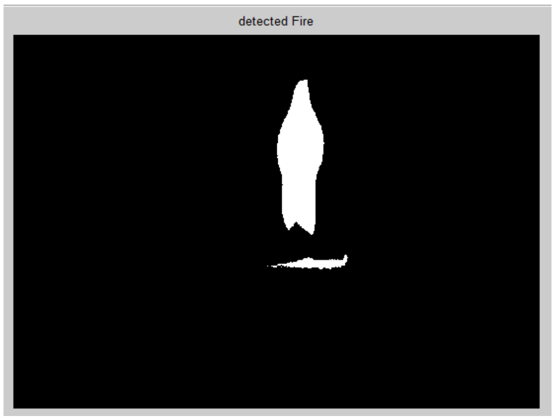


Figure III. 9 Filling the connected component regions

Visualizing the detection and computing the center of fire

To prove the presence of fire we draw a rectangle around the contour of the flames. A ‘+’ sign in the center of it. This is done by first getting the label matrix of the binary image, in which different regions of connect components are identified.

Then getting properties of those regions such as the area, centroid and the BoundingBox so that we can identify the center of all those connected component regions, and we compute the mean contour of the regions to draw our rectangle, as in the figure III.10.

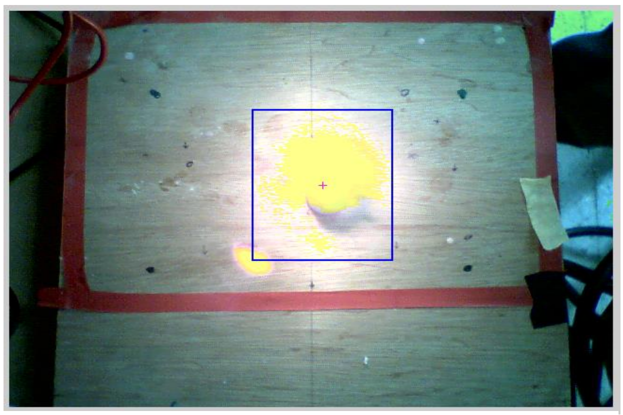


Figure III. 10 Blue box for contour of fire, '+' for center of fire.

After the determination of the center of fire, the pixel coordinates of the center are passed to the **angle** function which translates these coordinates to angles for the motor, based on geometrical calculations. The figures III.12 and III.11 bellow shows an example of how the angles from Matlab directed the cannon towards the source of fire.

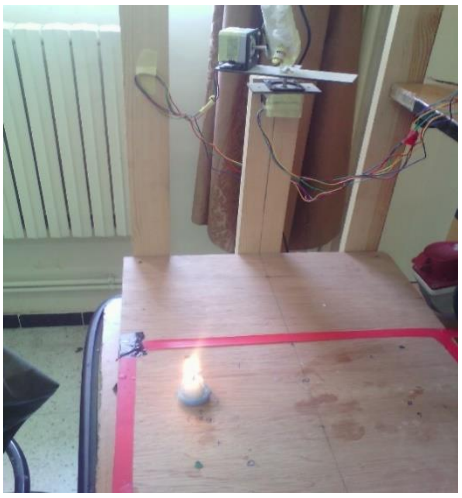


Figure III. 12 Angles directing the cannon to fire source

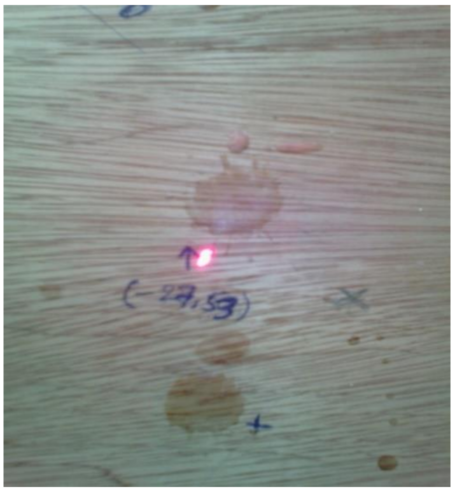


Figure III. 11 Arrow: position of fire & laser dot from cannon

III.1.2 Algorithm operation Testing

Algorithm is tested using two different fire sources of class A by changing the distance and angle of exposure to the webcam. The results and explanation are shown in the following:

Testing candle light:

- Exposing a candle fire to webcam, the fire was successfully detected in the range [0 ; 2] meters, in daylight, as seen in figure III.13, same outcome when tested in fluorescent light.

  In the dark The range of detection increases in the dark [0 ; 2.75] meters.

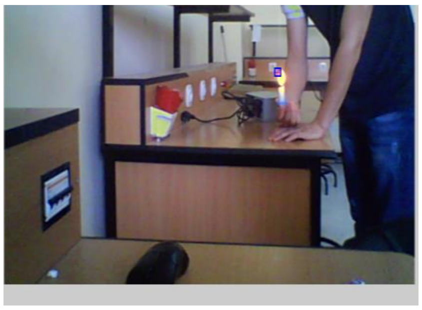


Figure III. 13 Fire barely detected in distance of about 2m.

- Expanding the size of fire by combining the fire of two candles simultaneously, fire was successfully detected in the range [0 ; 2.3] meters, in day and fluorescent light.

  In the dark the range increases to [0 ; 3] meters because of absence of other high intensity lights that shadow the light of fire.

The detection stopped in both cases because the area of the flame became less than (20x24) pixel are in our frame, noting the frame resolution is medium (640x480). For a flame to be detected in a scene it has to occupy a mean area greater than (24x24) pixels.

Testing paper fire:

- Now we change the source of fire to paper, we found that even though the flames from paper where larger than the candle fire, but the range of detection is not that big, detection success range is [0 , 2] meters in daylight, and fluorescent light, as in figure III.14.

This is due to the fact that the flames captured by webcam are blurred by the dark smoke, thus reducing the intensity light emitted from flames to webcam and the YCbCr rules will not apply thus fire will not be detected.

We noticed that range increases significantly in the dark for the same paper size burned in dark: [0 , 3] meters.

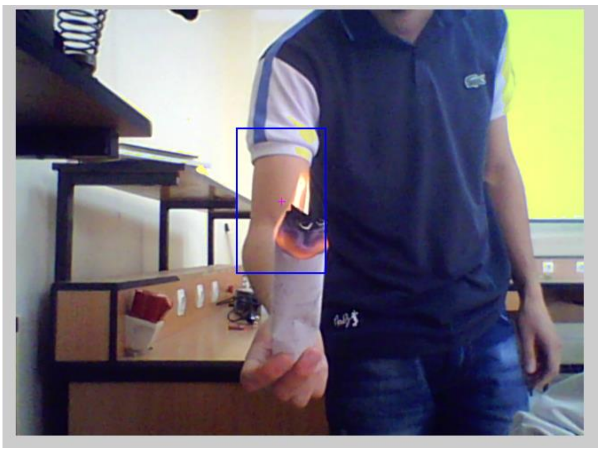


Figure III. 14 Paper fire detected

Testing exposure angle:

- Testing different exposure angles in our implemented hardware: very accurate detection of fire was registered for locations not near the boundaries of the webcam’s scope. Scope is the visual location covered by camera. The Fire (‘+’) center deviates slightly when the testing fire is near the boundary of the scope. This due to fact that a portion of light intensity is missing from the frame (not covered by scope).

  Calculating only the light present in the frame results in a center not exactly over the fire. Leading to some tolerable deviation in the cannon hitting exactly the fire.

  The detection is not affected whether its daylight, fluorescent or dark.

- Finally we test the accuracy of our computed angles using the FPGA and the laser beam, we found that for test-fire in the scope of camera the laser pointer is in the vicinity of a diameter of [0 ; 3] cm from center of fire, and for test-fire outside the scope of camera laser pointer is in the vicinity of a diameter of [0;5]cm.

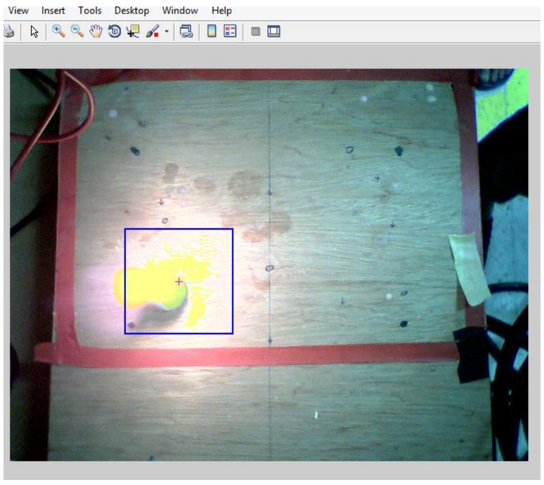


Figure III. 15 Accurate detection when fire is in scope of webcam

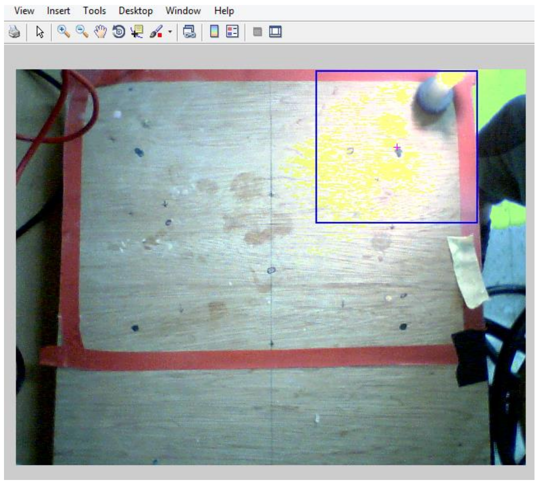


Figure III. 16 Poor detection for fires out of scope

.

Testing calculated angles

Function **angle** Updates angles of the cannon from current position to the new position, if the fire has displace. Tested for displacement of different positions of fire using the laser pointer it showed accuracy of the function. But due to the some measurements error in the first place, accumulated error for 4 to 5 fire displacement can be in the range of [0 , 5] cm.

Execution time:

Using clocking functions in Matlab, the algorithm processes one frame in around 0.31 second, or three frames each second. Which is a good time, where processing one frame in five to ten seconds is acceptable for the detection and then extinction of fire.

III.1.3 Conclusion on Matlab algorithm:

The Matlab algorithm has proved accurate results of detection and location of fire in cases where the fire is in the scope of the camera. In this case the laser pointer is in a diameter less than 3 cm from the position of test fire. If we apply water instead of laser sure it is going to be poured on the fire. The system can cope with different light intensities of the environment (Day light, dark, fluorescent light) also copes with displacement of fire, it updates the fire position from current cannon position to new fire positions.

III.2 Hardware Implementation & results

In this section of project, the fire extinguishing system is tested within a real implementation. Through two main steps:

1- Testing each element separately.

2- Testing the whole project.

III.2.1 The flame detector module

We have used a 1-channel flame detector module, in order to check the availability of fire source via emitted infra-red radiation.

Figure III. 17 Flame sensor module

The flame sensor was very responsive to the light of candle, in a maximum range of one meter from the source. The module has a digital output of '1', when it is disabled, and a '0' output when it is triggered.

III.2.2 The bipolar stepper motor components

As it is shown below in figure, both horizontal and vertical stepper motors are well fixed to a suitable wood support. And they are localized at their reference position.

A laser canon is then installed to the vertical side, in order to target the fire location.

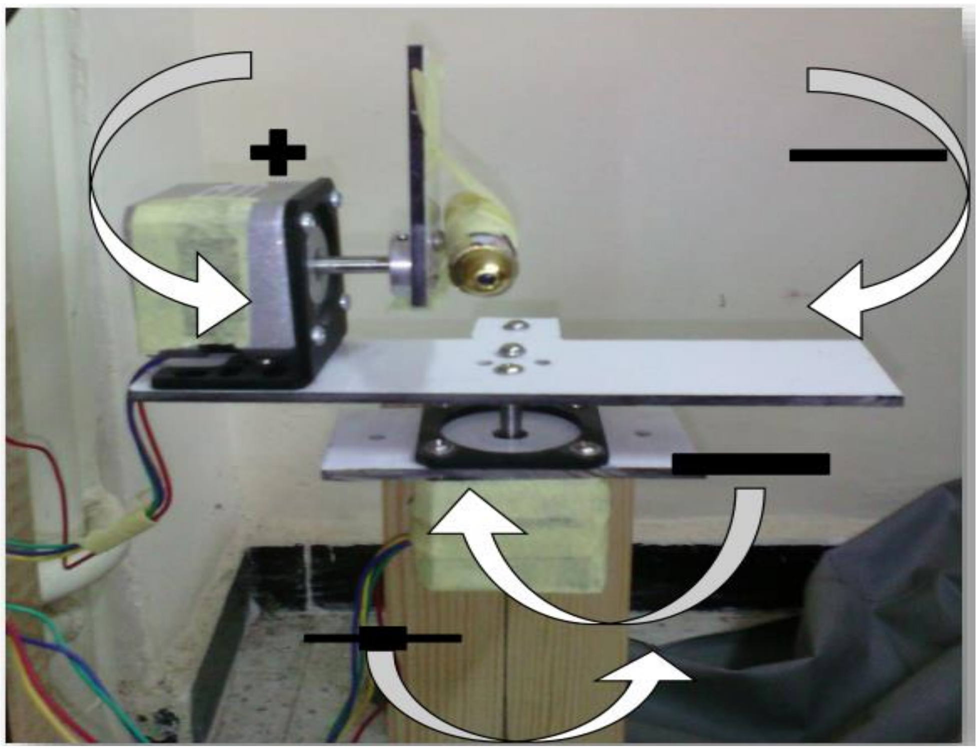


Figure III. 18 Stepper motors at reference position, and sign orientation.

Each motor has been controlled to rotate either in positive (counter clockwise) direction, or in negative (clockwise) direction, based on the received data from MATLAB, refer to figure III.17.

III.2.3 Water level indicator

The water level sensor has been simulated using a potentiometer to simulate variable values from the water sensor. Chips 74LS244, 74HC244 are used for connecting digital ADC output to FPGA. Levels of water are set according to the value from a potentiometer.

It begins from level 0 to level 7. Level 0 and 1 means that the tank is low on extinguishing agent, thus triggering the alarm (turning LEDs on). While the rest of levels from 2 up to 7 means that tank is filled and no alarm is released.

III.2.4 Pump

The pump has been simulated using a 12V-DC motor of brushless fan. This fan is used for shooting water. It is activated when the flame sensor is triggered. The fan is interfaced with FPGA using a relay, in order to drive the 12V-DC motor inside it.

Figure III. 19 DC motor

III.2.5 Alarm:

The Alarm is simulated using LEDs.

This LED alarm is activated in two cases:

- The water tank is at very low level.
- A fire threat has been confirmed by the flame detector

III.3 Hardware System prototype:

The hardware system prototype is shown below in Figure III.21.

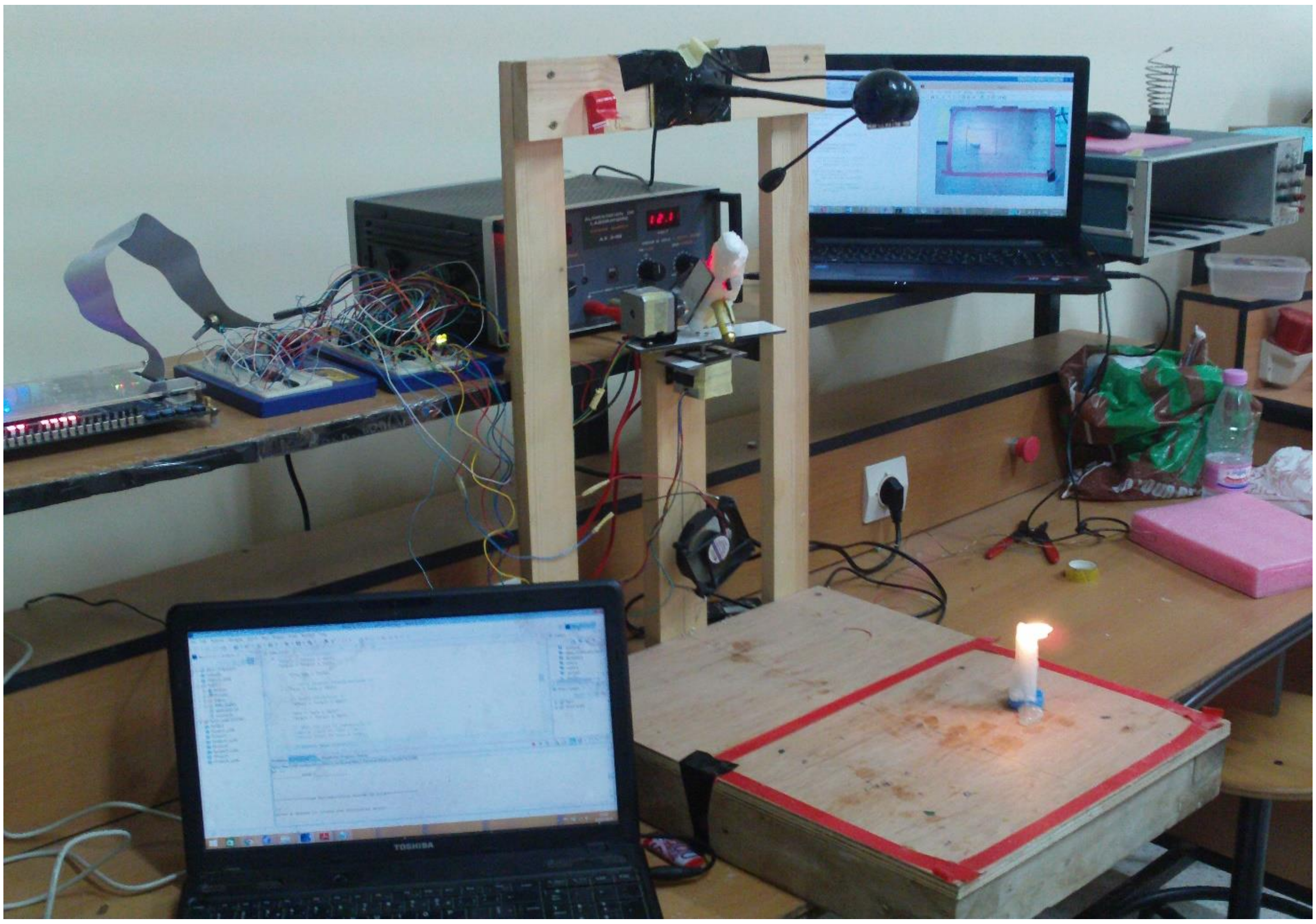


Figure III. 20 Hardware system prototype

We have been able so far to use the Keyboard of the computer that is running NIOS II IDE and Matlab, to test the operation of our system. While other methods are still under test to send the angles automatically to the NIOSII processor through the serial JTAG connection.

The Matlab program displays the angles of location of fire, these angles are then typed manually into Nios II IDE terminal to be read. The C-program uses this data and instruct the Steppers motors through FPGA, to move to desired location.

Several cases have been tried to test the correct operation of the system:

Testing water level

- Case 1: the angles data are available and flame detector is used, but the tank level is 1 ➔

  Stepper motors do not rotate but a warning alarm is enabled.

  Case 2: the angles data are available, level is 7, and flame detector is used ➔

  Stepper motors rotate, brushless fan and warning alarm are both enabled.

Testing flame detector

- Case 3: angles are some position water level is 6 and flame detector is used ➔ brushless fan and warning alarm are both enabled.

- Case 4: angles are some position, and water level is 4, but flame detector is not used ➔ nothing works.

The table below summarizes the precedent cases:

Table III.1 Test cases

Case characteristics:	Water level	Stepper motor 1	Stepper motor 2	Flame detector	Brushless fan	Warning alarm
• water level = 1 • angle 1: 20° • angle 2: 30° • with a flame detector	Level 1	Off	Off	1	Off	On
• water level = 7 • angle 1: -25° • angle 2: 30° • with flame detector	Level 7	On	On	0	On	On
• water level = 6 • angle 1: -25° • angle 2: 30° • without flame detection	Level 6	On	On	1	Off	Off

Results and discussion:

- The implemented system successfully detect the presence of fire.
- Angles are then computed and transferred to the FPGA.
- Angles precision of stepper motors using a half step sequencing.
- Angles are tested using the cannon and have shown precise positioning.
- The system is working only when the extinguishing agent is sufficient, otherwise a warning alarm is raised. Alarms are simulated using LEDs.
- The warning alarm is generated either if the tank gets almost empty or when there is a detected fire.
- A fan DC motor is used to simulate a pump.
- The pump and warning alarms will not be enabled unless a flame is detected by flame sensor, as in this example of false alarm from MATLAB. Hence, any false alarms is avoided from MATLAB algorithm.

- To reduce angle precision errors, we use half step sequencing instead of the full step one. Good results are obtained. Results were as expected within the tolerated range of errors: The rotating angle error, which was less than 0.5°.
- Another problem was in the insufficient size of the on-chip memory inside the FPGA. This has been solved by the 512-Kbyte SRAM of the DE2 board instead of the limited size on chip memory.
- A delay function has to be implemented so that synchronization between the frequency of the microprocessor and the speed of stepper motors do not overlap, avoiding step loss.

Conclusion on hardware implementation:

The stepper motors rotated with a small error using half step. Accurate positioning was registered. The flame sensor worked properly in accordance with the positioning of the cannon. Detection from sensor was correct for fires not far than 1 meter. The alarm and the water pump operated correctly in cases of an alarm. The system took few seconds to move to the fire and take action according to the objectives set at the beginning of this project.

Conclusion and future work

The automatic fire suppression system designed has proved its robustness in terms of effective detection and avoiding false alarms. The video processing algorithm have succeeded to precisely locate the fire from the video stream and order the cannon to aim at it and extinguish it, all in a matter of seconds. The aiming of the cannon gave satisfying results where a very accurate aiming is not required. The adaptation of the system to the displacement of fire and the different light environment was very good. This system really provide a cheap alternative to the nozzle system in addition to low false alarms and suppression of fire only where fire is located.

There's still a lot to improve in the system, starting by the algorithm, where we suggest a smoke detection algorithm using the video stream should be developed, to work in parallel with our system and provide a backup where our algorithm fails. The algorithm should be extended to detecting other fire classes, where the color and intensity of light emitted change accordingly, and can decide which fire class it is and which fire agent should be used. So a variety fire agents should be used with the cannon.

The FPGA board is powerful when it comes to implementing an embedded system. The whole process can be implemented fully inside an FPGA including the video acquisition and the fire detection using image processing algorithms. Based on this approach, the FPGA controls all the external and internal operations of the system. In this way we remove the computer from the system and it becomes as one independent unit, and hence a standalone embedded system. However, this will introduce a number of challenges including video acquisition, memory limitations, and the lack of image processing libraries like OpenCV.

Since we said providing a replacement for the nozzle system, means large scale implementation of the system, a centralized system is adequate in which a server is used to collect data from scattered cameras and sensors, and send data to PLCs to drive the cannons. Many cannons means a power grid should be studied and supplied. Finally for better effective detection high resolution cameras should be used for large area coverage, in combination with long range flame sensors capable of sensing up to 100m.

References

[1] M.-G. Leonardo, S.-P. Gabriel, N. Mariko, T.-M. Karina, P.-M. Hector and R.-C. Luis, "An Early Fire Detection Algorithm Using IP Cameras," 3 May 2012..

[2] P. Battise and A. B. S, "A FAST AND EFFICIENT VISION BASED APPROACH FOR DETECTION OF FIRE IN REAL TIME SCENARIO," Agarkar, E&Tc dept. SRESCOE, Kopargaon, India, june 2015.

[3] E.Memane and V.S.Kulkarni, "A Review on Flame and Smoke Detection Techniques in Video's," College of Engineering, Pune, India, February 2015.

[4] wikipedia, "Fire," [Online]. Available: https://en.wikipedia.org/wiki/Fire. [Accessed 01 05 2016].

[5] U. o. t. Austin, "Fire prevention services," [Online]. Available: https://www.utexas.edu/safety/fire/extinguishers/abcs.html. [Accessed 01 05 2016].

[6] "Automatic Fire Suppression," [Online]. Available: https://en.wikipedia.org/wiki/Automatic_fire_suppression. [Accessed 15 05 2016].

[7] T. L. S. G. Fire Equipment Manufacturers' Association (FEMA, "An Overview of Pre-Engineered Fire Suppression Systems and Industry Changes That May Affect Your Property.," june 2007.

[8] C. G. Raphael, E. W. Richard and S. L. Eddins, Digital Image Processing Using MATLAB, Gatesmark Publishing., 2009.

[9] wikipedia, "Field-programmable_gate_array," [Online]. Available: https://en.wikipedia.org/wiki/Field-programmable_gate_array. [Accessed 05 05 2016].

[10] wikipedia, "Altera_Quartus SOPC_Builder," [Online]. Available: https://en.wikipedia.org/wiki/Altera_Quartus#SOPC_Builder. [Accessed 2016 06 2016].

[11] Altera, "Introduction to the Altera Nios II Soft Processor," [Online]. Available: https://www.altera.com/, Introduction to the Altera Nios II Soft Processor. [Accessed 01 05 2016].

[12] B. Earl, "What is a Stepper Motor?," [Online]. Available: https://learn.adafruit.com/all-about-stepper-motors/what-is-a-stepper-motor. [Accessed 25 04 2016].

[13] github, " Github repository for Grove Flame Sensor Library," [Online]. Available: https://github.com/Seeed-Studio/Grove_Flame_Sensor. [Accessed 25 05 2016].

[14] [Online]. Available: http://www.seltechfire.ca/sprinkler-and-suppression. [Accessed 02 05 2016].

[15] [Online]. Available: http://stseg.net/Home/index.php?option=com_content&view=article&id=31&Itemid=79. [Accessed 02 05 2016].

[16] [Online]. Available: http://proshieldfireandsecurity.com/suppression-systems/wet-chemical-systems/. [Accessed 02 05 2016].

[17] [Online]. Available: http://tank-accident.blogspot.com/2015/05/blog-post_21.html. [Accessed 02 05 2016].

[18] "fire suppression system CT," fire suppression system CT, [Online]. Available: http://firesuppressionsystemct.com. [Accessed 01 05 2016].

[19] [Online]. Available: https://twitter.com/sm123489. [Accessed 01 05 2016].

[20] [Online]. Available: https://twitter.com/sm123489. [Accessed 01 05 2016].

[21] "fire suppression system ct," fire suppression system ct, [Online]. Available: http://firesuppressionsystemct.com. [Accessed 02 05 2016].

[22] [Online]. Available: http://ruthburke.photoshelter.com/image/I0000mw1OhFwnRTk. [Accessed 10 04 2016].

[23] [Online]. Available: http://www.esng.dibe.unige.it/deeds/LearningMaterials/LM/Tutorials/DE2/Index.htm. [Accessed 10 05 2016].

[24] A. Corporation, "Designing with the Nios II Processor and SOPC Builder," Altera Corporation, 2009.

Table of Contents

LIST OF FIGURES

LIST OF TABLES