A FPL Bioinspired Visual Encoding System to Stimulate Cortical Neurons in Real-Time

Leonel Sousa[1], Pedro Tomás[1], Francisco Pelayo[2],
Antonio Martinez[2], Christian A. Morillas[2], and Samuel Romero[2]

[1] Dept. of Electrical and Computer Engineering, IST/INESC-ID, Portugal
las@inesc-id.pt, pfzt@sips.inesc-id.pt
[2] Dept. of Computer Architecture and Technology, University of Granada, Spain
fpelayo@ugr.es, {amartinez, cmorilas, sromero}@atc.ugr.es

Abstract. This paper proposes a real-time bioinspired visual encoding system for multielectrodes' stimulation of the visual cortex supported on Field Programmable Logic. This system includes the spatio-temporal preprocessing stage and the generation of varying in time spike patterns to stimulate an array of microelectrodes and can be applied to build a portable visual neuroprosthesis. It only requires a small amount of hardware which is achieved by taking advantage of the high operating frequency of the FPGAs to share circuits in time. Experimental results show that with the proposed architecture a real-time visual encoding system can be implemented in FPGAs with modest capacity.

1 Introduction

Nowadays, the design and the development of visual neuroprostheses interfaced with the visual cortex is being tried to provide a limited but useful visual sense to profoundly blind people. The work presented in this paper has been carried out within the EC project "Cortical Visual Neuroprosthesis for the Blind" (CORTIVIS), which is one of the research initiatives for developing a visual neuroprosthesis for the blind [1, 2].

A block diagram of the cortical visual neuroprosthesis is presented in fig. 1. It includes a programmable artificial retina, which computes a predefined retina model, to process the input visual stimulus and to produce output patterns to approximate the spatial and temporal spike distributions required for effective cortical stimulation. These output patterns are represented by pulses that are mapped on the primary visual cortex and are coded by using Address Event Representation [3]. The corresponding signals are modulated and sent through a Radio Frequency (RF) link, which also carries power, to the electrode stimulator. This project uses the Utah microelectrode array [4], which consists on an array of 10×10 silicon microelectrodes separated by about 400 μm in each orthogonal direction (arrays of 25×25 microelectrodes are also considered). From experimental measures on biological systems, it can be established a time of 1 ms to "refresh" all the spiking neurons, which means an average time slot of 10 μs dedicated to each microelectrode. The RF link bandwidth allows communication

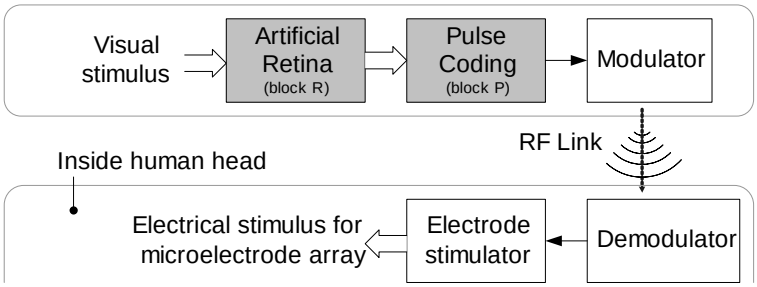

Fig. 1. Cortical visual neuroprosthesis

at a bit-rate of about 1 Mbps, which means an average value of 10 kbps for each microelectrode in a small size array of 10×10 electrodes or about 1.6 kbps for the 25×25 microelectrode array.

This paper addresses the design of digital processors for implementing the shady blocks in fig. 1 in Field Programmable Logic (FPL) technology. The model of the retina adopted is a complete approximation of the spatio-temporal receptive fields' characteristic response of the retina ganglion cells (block R). The neuromorphic pulse coding is based on a leaky integrate-and-fire model of spiking neurons and on the Address Event Representation (AER), that communicates information about the characteristics of spikes and addresses of target microelectrodes without timestamps (block P). The architecture of the system has been designed having in mind the specifications of the problem referred above and the technical characteristics of nowadays Field Programmable Gate Array (FPGA) devices. Experimental results show that a complete artificial model that generates neuromorphic pulse-coded signals can be implemented in real-time even in FPGAs with low-capacity.

This paper is organized as follows. The architecture for modeling the retina and for coding the event lists that will be carried out to the visual cortex is presented in section 2. Section 3 reports the computational architectures designed for implementing the retina model in FPL, discussing their characteristics and suitability to the technology. Section 4 presents experimental results obtained by implementing R and P blocks on a FPGA and section 5 concludes the paper.

2 Model Architecture

The neuron layers of the human retina perform a set of different tasks, which culminate in the spiking of ganglion cells at the output layer [5]. These cells have different transient responses, receptive fields and chromatic sensibilities. The system developed in this paper implements the full model of the retina, which includes all the processing layers, plus the protocol for carrying the spikes to the visual cortex in a serial way through a RF link.

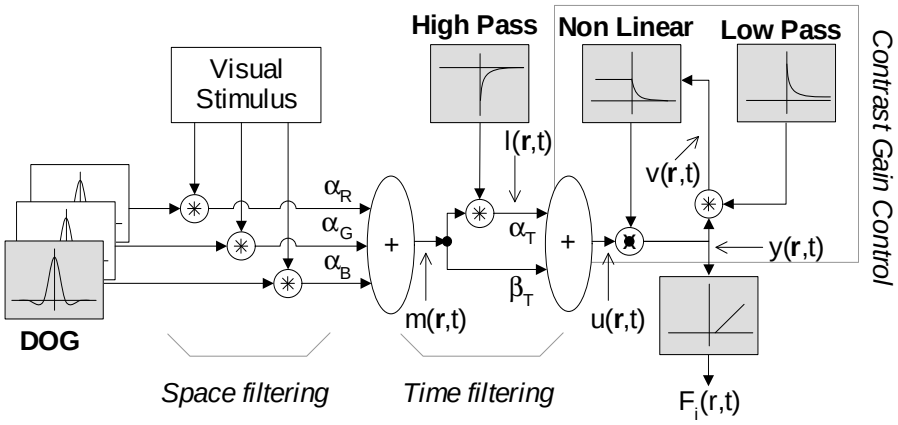

Fig. 2. Retina early layers

The two main blocks of the system in fig. 1 perform the following tasks: block **R**, the spatiotemporal filtering of the stimulus visual input, with contrast gain control and a rectifier circuit for computing the firing rate (fig. 2); block **P**, a neuromorphic pulse coding block which also implements the protocol used to communicate event lists without timestamps (fig. 3).

2.1 Retina early layers

The retina computational model used is based on the research published in [6], but it has been extended to the chromatic domain by considering independent filters for the basic colors. Fig. 2 presents the model architecture. It includes a spatial filter for contrast enhancement in the various color components which output is combined with programmable weights to reproduce receptive fields stimulated by specific color channels. A Difference of Gaussians (DoG) is used to filter the stimulus in space:

$$DOG(\mathbf{r}) = \frac{\mathbf{a_+}}{\mathbf{2\pi\sigma^2}}\mathbf{e}^{-\frac{\mathbf{r^2}}{\mathbf{2\sigma^2}}} - \frac{\mathbf{a_-}}{\mathbf{2\pi\beta^2\sigma^2}}\mathbf{e}^{-\frac{\mathbf{r^2}}{\mathbf{2\beta^2\sigma^2}}} \tag{1}$$

where a_+ and a_- represent the relative weights of center and surround, respectively, and σ and $\beta\sigma$ ($\beta > 1$) are their diameters.

A high-pass temporal filter with the following impulse response is applied to the input signal already filtered in space:

$$h_{HP}(t) = \delta(t) - \alpha H(t) e^{-\alpha t} \tag{2}$$

where $H(t)$ represents the Heaviside step function and α^{-1} is the decay time constant of the response. In this paper, the bilinear approximation was applied to derive a digital version of the filter represented in the Laplace domain [7]. It

leads to a first order Infinite Impulse Response (IIR) digital filter which can be represented by equation 3.

$$l[n] = b_{HP} \times l[n-1] + c_{HP} \times (m[n] - m[n-1]) \tag{3}$$

The relevance (weight) of the time response of a particular receptive field is also programmable in the model presented in fig. 2. The resulting activation $u(\mathbf{r}, t)$ is multiplied by a Contrast Gain Control (CGC) modulation factor $g(\mathbf{r}, t)$ and rectified to yield the ganglion cells firing rate response to the input stimulus.

The CGC models the strong modularity effect exerted by stimulus contrast. The CGC non-linear approach is also used in order to model the "motion anticipation" effect observed on experiments with a continuous moving bar [8]. The CGC feedback loop involves a low-pass temporal filter with the following impulse response:

$$h_{LP}(t) = Be^{-\frac{t}{\tau}} \tag{4}$$

where B and τ define the strength and constant time of the CGC. The filter is computed in the digital domain, by applying the same approximation as for the high-pass filter, by equation 5.

$$v[n] = b_{LP} \times v[n-1] + c_{HP} \times (y[n] + y[n-1]) \tag{5}$$

and the output of the filter is transformed into a local modulation factor (g) via the non-linear function:

$$g(t) = \frac{1}{1 + [v(t) \cdot H\big(v(t)\big)]^4} \tag{6}$$

The output is rectified by using the function expressed in eq. 7:

$$F_i(\mathbf{r}, t) = \psi H\big(y(\mathbf{r}, t) + \theta\big)[y(\mathbf{r}, t) + \theta] \tag{7}$$

where ψ and θ define the scale and baseline value of the firing rate $f_i(\mathbf{r}, t)$.

The system to be developed is fully programmable and typical values for all parameters of the model are found in [6]. The model has been completely simulated in MATLAB, by using the *Retiner* environment for testing retina models [9].

2.2 Neuromorphic pulse coding

The neuromorphic pulse coding block converts the continuous-varying time representation of the signals produced in the early layers of the retina into a neural pulse representation. In this new representation the signal provides new information only from the moment a new pulse begins. The adopted model is a simplified version of an integrate-and-fire spiking neuron [10]. As represented in fig. 3, the neuron accumulates input values from the respective receptive field (output firing rate determined by retina early layers) until it reaches a given threshold. Then it fires and discharges the accumulated value. A leakage term is included to force

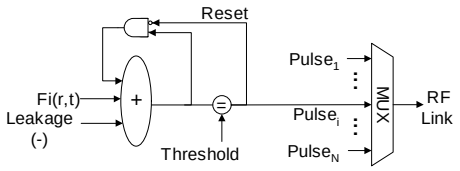

Fig. 3. Neuromorphic pulse coding

the accumulated value to diminish for low or null input values. The amplitude and duration of pulses are then coded by using AER, which is represented in a simplified way in fig. 3 by a multiplexer, to be sent to the addressed microelectrodes via the RF link. An event consists on an asynchronous bus transaction that carries the address of the corresponding microelectrode and is sent at the moment of pulse onset (no timestamp information is communicated). An arbitration circuit is required in the sender side and the receiver has to be listening to the data link with a constant latency.

3 FPL Implementation

This section discusses the implementation in FPL technology of the visual encoding system presented in the previous section. The usage of FPL technology will allow changing the model parameters without the need of projecting a second circuit. This has special importance since different patients have different sight parameters therefore requiring adjustments to the artificial retina. FPL may also allow changing model blocks if new information on how visual stimulus are processed reveals the need to introduce new filters.

For the design of the system, different computational architectures were considered both for the retina early layers and to the neuromorphic coding of the pulses. These architectures lead to implementations with different characteristics, in terms of hardware requirements and speed. The scalability and programmability of the system are also relevant aspects that are taken into account for FPL implementations.

3.1 The retina early layers

To implement the retina early layers there can be multiple approaches which involve different hardware requirements, so the first step would be to analyze the necessary hardware to build the desired processor. Assuming a typical convolutional kernel of 7×7 elements for the DoG spatial filter and that high-pass and low-pass temporal filters are computed by using the difference equations 3 and 5, respectively, then 53 multiplications and 52 additions per image cell are required for just one spatial channel. For a matrix of 100 microelectrodes, the hardware

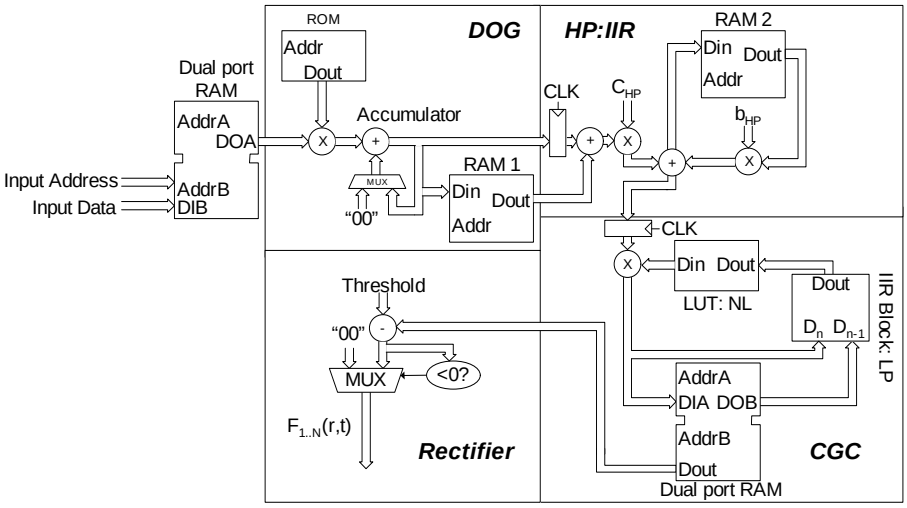

Fig. 4. Detailed diagram for the retina computational architecture.

required by a fully parallel architecture makes it not implementable in FPL technology. Therefore, the usage of the hardware has to be multiplexed in time, but the temporal restriction of processing the whole matrix with a frequency up to 100Hz must also be fulfilled. This restriction is however wide enough to consider the processing of cells with a full multiplexing schema inside each main block of the architecture model presented in section 2: DoG spatial filter, High-Pass temporal filter and Contrast Gain Control.

The architectures of these blocks can then be described as shown in figure 4, where each main block includes one or more RAM blocks to save processed frames in intermediate points of the system in order to compute the recursive time filters. At the input, dual-port RAM for the three color channel is used, while ROM is used to store the three coefficient tables for the spatial DoG filters. The operation of the processing modules is locally controlled and the overall operation of the system is performed by integrating the local control in a global control circuit. The control circuit is synthesized as a Moore state machine, because there is no need of modifying the output asynchronously with variations of the inputs. All RAM address signals, which are unconnected in the figure, are generated by the control circuit.

The DoG filter module calculates the 2D convolution between the input image and the predefined matrix coefficient stored in ROM. It operates in a sequential method, where pixels are processed one at a time in a row major order and an accumulator is used to store the intermediate and final results of the convolution. The local control circuit stores the initial pixel address, row and column, and then successively addresses all the data, and respective filtered coefficients stored in the ROM, required for the calculus of the convolution. After multiply-

ing each filter coefficient the resulting value is successively accumulated. When the calculus is finished for a given pixel, the value is sent both to the internal RAM1 and to the next high-pass (HP) filter module. This module receives as operands the space filter results for the actual and the previous images ($m(\mathbf{r}, n)$ and $m(\mathbf{r}, n-1)$) and the previous filter output $l(\mathbf{r}, n-1)$, which is stored in the RAM2. The filter is computed by first adding both $m[n]$ and $m[n-1]$ inputs, then multiplies the sum by the coefficient c_{HP} and the product is added with the previous filter output result $l[n-1]$ multiplied by b_{HP}. The result is both stored in the RAM2 and communicated to the next processing module that corresponds to the CGC. This module consists on a low pass filter (eq. 5) and a non-linear function (eq. 6) which is computed by using a lookup table stored in a RAM block. The low-pass filter circuit is not detailed in the figure because it is similar to the high-pass one. The last processing module is a rectifier which cuts the signal whenever its value decreases below a predefined threshold. It is implemented by a simple binary comparator whose output is used to control a multiplexer.

The overall circuit operates in a 3 stage pipeline corresponding to each one of the main processing blocks. Only the first pipeline stage require a variable number of clock cycles depending on the dimension of the filter kernel–49 cycles for a 7×7 kernel. Each of the two other pipeline stages is solved in a single clock cycle.

3.2 Neuromorphic pulse coding

Fig. 5 shows two processing modules *i)* for pulse generation and its representation and *ii)* to arbitrate the access to the serial bus (the input to the RF modulator in figure 1). This block is connected to the retina early layers through a dual port RAM (CGC block in fig. 4) where one writes data onto one port and the others reads it from the other.

The pulse generation circuit, which converts from firing rate to pulses, can be seen as a simple Digital Voltage Controlled Oscillator (DVCO) working in a two clock cycle stage pipeline. In the first stage the input firing rate is added to the accumulated value. In the second stage a leakage value is subtracted and, if the result is bigger than the threshold, a pulse is fired and the accumulator returns to the zero value (see fig. 5). Note that in this architecture the accumulator circuit is made with a RAM, since it corresponds to a single adder and multiple accumulator registers for the different microelectrodes.

AER was used to represent the pulses while the information is serialized. In a first approach, the architecture consists of an arbitration tree to multiplex the onset of events (spikes) onto a single bus. In this tree, we check if one or more inputs had a pulse to be sent and arbitrate the access to the bus trough the request/acknowledge signals in the tree [3]. This tree consists of multiple subtrees, where registers can be used for buffering the signals between them. This architecture is not scalable, since it requires a great amount of hardware and is not a solution when arrays with a great number of microelectrodes are used (see section 4). To overcome this problem, and since the circuit for pulse generation

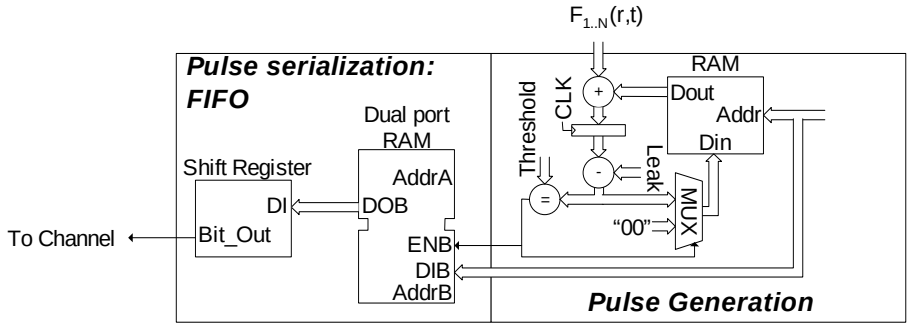

Fig. 5. Block diagram of the circuit for coding the pulses and to implement the address event protocol.

is intrinsically sequential, a First In First Out (FIFO) memory is used to register the generated spikes for the microelectrodes. Only one pulse is generated in a clock cycle and information about it is stored in the FIFO memory because requests may find the communication link busy. This implementation has the advantage of not increasing the hardware resources required with the increase in the number of microelectrodes, but its performance is more dependent of the channel characteristics since it increases the maximum wait time to send a pulse.

The FIFO is implemented by using a dual port data RAM, where input is stored in one port and, at the other port, the pulse request is sent to the channel. Also to avoid overwriting when reading data from the FIFO a shift register is used to interface the data output of the RAM with the communication channel.

4 Experimental Results

The visual encoding system was described in VHDL and exhaustively tested on a DIGILAB II board supported on a XILINX SPARTAN XC2S200 FPGA [11] with modest capacity. The synthesis tool used was the one supplied by the FPGA manufacturer, the ISE WebPACK 5. This FPGA has modest capacities, with 56 kbit of block-RAM and 2352 slices, corresponding to a total of $200,000$ system-gates. The functional validation of the circuits was carried out by comparing the results obtained with MATLAB Retiner [9].

For the test and experimental results presented in this section, only one space filter (DoG) is considered and the input signal is represented in 8-bit grayscale. However, internally 12-bit signals are used in order to reduce discretization errors. The considered dimension of the microelectrode array is 100.

Analyzing the results in table 1 it is clearly seen that the retina early layers do not require many FPGA slices but occupies a significant amount of memory. The synthesis of the circuit for an array of 1024 microelectrodes shows a similar percentage of FPGA slice occupation but it occupies all the 14 RAM blocks supplied by the SPARTAN XC2S200. In terms of time restrictions, the solutions

Table 1. Retina encoding system implemented on a SPARTAN XC2S200 FPGA

Block	**Number of microelectrodes**	**Slice Occupation**	**Maximum clock frequency**	**Number of RAM blocks used**
Retina early layers	100	18%	47MHz (49 cc*)	6
Pulse Generation	100	2%	51MHz	5
AER	100	9%	99MHz	0
Registered Tree	256	17%	98MHz	0
	512	37%	93MHz	0
AER	100	10%	64MHz	0
Unregistered Tree	256	21%	63MHz	0
	512	45%	57MHz	0
FIFO AER	**	5%	75MHz	1

* 49 clock cycles are required for processing a pixel
** not dependent of the number of micro-electrodes

works very well since it can process the array of input pixels in a short time, about 0.1ms for 100 pixels, which is much lower than the specified maximum of 10ms. In fact this architecture allows to process movies at frame rate of 100 Hz and a resolution of 10000 pixels without increasing the number of slices used.

The analysis of the AER translation method as however different aspects. If considered a typical matrix of 100 microelectrodes the tree solution (with or without intermediate registers) is a valid one as the resource occupation is low. However, the increase of the number of microelectrodes implies the usage of a great amount of hardware and this solution becomes impracticable for FPL technology. In that case, the FIFO based solution proposed in this paper has the great advantage of accommodating a great number of electrodes with a small amount of hardware and just one RAM block. With a channel bandwidth of about 1 Mbps, the FIFO based AER circuit does not introduce any temporal restrictions in the firing rate. Even operating at a lower clock frequency than the admitted by the retina early layers circuit, e.g. 40 MHz, the FIFO based circuit is able to attend 100 requests in about 2.5 μs which is much less than the value of 1 ms initially specified and also much less than the time required to send the information through the channel.

In table 1 results are individually presented for the retina early layers and for the neuromorphic pulse coding circuits. The overall system was also synthesized and implemented in a SPARTAN XC2S200 FPGA. A clock frequency greater than 40 MHz is achieved by using about 25% of the slices and 12 of the total of 14 RAM-blocks, for 100 microelectrodes.

5 Conclusions

This paper proposes a computational architecture for implementing a complete model of an artificial retina in FPL technology. The system is devised to stim-

ulate a number of intra-cortical implanted microelectrodes for a visual neuroprosthesis. It performs a bio-inspired processing and encoding of the input visual information. The proposed processing architecture is designed to respect the constraints imposed by the telemetry system to be used, and with the requirement of building a compact and portable hardware solution.

The synthesis results demonstrate how the adopted pipelined and time multiplexed approach makes the whole system fit well on a relatively low complexity FPL circuit, while real-time processing is achieved. The use of FPL is also very convenient to easily customize (re-configure) the visual pre-processing and encoding system for each implanted patient.

Acknowledgements

This work has been supported by the European Commission under the project CORTIVIS ("Cortical Visual Neuroprosthesis for the Blind", QLK6-CT-2001-00279).

References

1. Cortica visual neuro-prosthesis for the blind (cortivis): http://cortivis.umh.es.
2. Ahnelt P., Ammermller J., Pelayo F., Bongard M., Palomar D., Piedade M., Ferrandez J., Borg-Graham L., and Fernandez E. Neuroscientific basis for the design and development of a bioinspired visual processing front-end. In *Proc. of IFMBE*, pages 1692–1693, Vienna, 2002.
3. Lazzaro J. and Wawrzynek J. A multi-sender asynchronous extension to the address event protocol. In *Proc. of 16th Conference on Advanced Research in VLSI*, pages 158–169, 1995.
4. Maynard E. The utah intracortical electrode array: a recording structure for potential brain-computer interfaces. *Elec. Clin. Neurophysiol.*, 102:228–239, 1997.
5. Wandell Brian. *Foundations of Vision: Behavior, Neuroscience and Computation*. Sinauer Associates, 1995.
6. Wilke S., Thiel A., Eurich C., Greschner M., Bongard M., Ammermuler J., and Schwegler H. Population coding of motion patterns in the early visual system. *J. Comp Physiol A*, 187:549–558, 2001.
7. Oppenheim A. and Willsky A. *Signal and Systems*. Prentice Hall, 1983.
8. Berry M., Brivanlou I., Jordan T., and Meister M. Anticipation of moving stimuli by the retina. *Nature (Lond)*, 398:334–338, 1999.
9. Pelayo F., Martinez A., Romero S., Morillas Ch., Ros E., and Fernández E. Cortical visual neuroprosthesis for the blind: Retina-like software/hardware preprocessor. In *Proc. of IEEE-EMBS International Conference on Neural Engineering*, Capri, 2003.
10. Gerstner W. and Kistler W. *Spiking Neuron Models*. Cambridge University Press, 2002.
11. XILINX. *Spartan-II 2.5V FPGA Family: Functional Description*, 2001. Product Specification.