UNIVERSITY OF OSLO
Department of Informatics

Computing visual disparity with temporal codes

Frode Bergh

Siv Ing Thesis

30 July 2002

PREFACE

The work presented in this diploma thesis, for the title of Siv.Ing., has been carried out at the Microelectronic Systems Group, Dept. of Informatics, University of Oslo. I started working on my thesis in September 2000.

I would like to thank my supervisors Philipp Häfliger and Tor Sverre Lande for their continuous advice and support during the whole process. I would also like to thank the other members of the group for giving constructive talks.

Last, but not least, I would like to thank Kristin Corneliussen Høslom. Without her undying support there would be no thesis at all. For this I am eternally grateful.

Blindern, July 2002

Frode Bergh

ABSTRACT

An electronic circuit is presented that computes the visual disparity between to input images. The output of the circuit, combined with the specification of the sensors/stereo camera that provides the images, can be used to find the distance between the sensors and the object photographed.

A neuromorphic circuit has been developed to solve the task at hand. The fresh approach of *this* circuit is to employ temporal coding. The first processing step is therefore to project the analog inputs into the time domain, i.e. into voltage spikes, the latency of which encodes the strength of the input. Thus the further comparison of pixel intensity can be done by asynchronous logic.

In theory the chip can compute the right visual disparity given two input images/stimuli. The actual aVLSI implementation has proven to have some limitations, but the chip can still compute the right visual disparity for most static images; after some initial fine-tuning of the biases. (The limitations are due to process variations in the production of the VLSI chip. Ways to reduce the effects of process variation are proposed.)

The circuit was implemented as a prototype on an AMS $0.6\mu m$ VLSI chip. The circuit takes 128 analog inputs, representing 2 images of 64 pixels. They are delivered as frequency encoded spike trains by a 7-bit AER (address event representation) bus. The output consists of 65 separate spike trains, each representing a disparity, multiplexed in a similar 7-bit AER bus. The frequency of spikes on each individual train represents the probability that the train corresponds to the right disparity.

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

1 INTRODUCTION

Extensive work has been done in neuroscience the last decades. Great advances have been made in the understanding of the nervous system of both animals and humans. Neuromorphic electronic systems try to benefit from this knowledge.

In short terms, neuromorphic electronic systems draw their inspiration from some biological solution to a problem. These neuromorphic systems are predominantly analogue, as opposed to almost all other modern electronic information-processing systems which are digital. As the main buzzword of electronics since the 70's has been "digital", it seems like neuromorphic engineering[1] is a step back in technology. The shear amount of digital circuits being produced keeps the prices low; and they get faster and more energy-efficient every day. The resolution, given in bits, steadily rises, making the circuits more and more accurate. So, why take the bother to develop analogue circuits?

Even though digital technology seems to evolve towards faster and faster circuits with virtually no limit, it still doesn't even come close to the efficiency and computation possibilities of the nervous system of even the simplest insect (Mead 1990: 1629). The dream of "artificial intelligence" that can rival the human brain actually seems increasingly far-fetched as we learn the possibilities and limitations of digital technology. Even the most efficient digital technology we can imagine today will be a factor 10 millions *less* efficient than the human brain (Mead 1990: 1629-1630). (It would require about 10 MW to process information at the same rate as a single human brain, which uses 1-2 W.) So, what to do? Well, we should not give up quite yet. The human brain has something to teach us that much are sure.

1.1 AREAS OF USE FOR NEUROMORPHIC ENGINEERING

Since the signals in the neuromorphic electronic systems are represented by relative values rather than absolute, they may not be suited for tasks like balancing check accounts. But, an ordinary desktop computer already brilliantly handles a task like that. The perfect choices for neuromorphic treatment are systems that have ill-defined/fuzzy input(s) that needs massive processing, often in parallel, to produce an approximate output. This is the kind of tasks the human brain most often handles, like for instance computing visual disparity.

> *" The disparity problem appears in stereo-vision: Our two eyes see the same scene from a slightly different angle. In order to perceive depth, we have to match objects in the picture from the left eye with objects in the picture from the right eye. We can then know, how far that object is away from us, since we know the difference in angle (disparity) at which it is seen from the left and right eye." (Häfliger 2000)*

[1] "Neuromorphic engineering" is the application of analogue CMOS VLSI technology to the fabrication of analogue electronic circuits that emulate real neural systems.

1.2 COMPUTATIONAL PRIMITIVES

Some aspects of the computations in the brain can be modeled with mathematical primitives like addition, subtraction, exponentials and integration. In digital systems these primitives are built of several AND, OR and NOT-gates; and the resolution of computation is limited to the number of bits in each value. The more bits, the more gates; and the complexity and energy dissipation grow steadily. In analogue electronics, Kirchoff's law of current implements addition and subtraction. The capacitance of a node integrates the current into it with respect to time. By using the subtreshold region of operation of the transistor, we get yet another computational primitive; the exponential relation between the input potential/voltage (V_{gs}) and the output current (I_{ds}). (Exponentials are not trivial to compute in digital technology). By using the transistor in the subtreshold region we also lower it's energy dissipation to within the shooting range of a single neuron (Mead 1990: 1630). The long-term memory of the brain can be modeled by the charge on a polysilicon node, which will hold the charge for years. (This is also the technology used in today's digital EPROM's.)

These primitives have several properties in common with the building blocks of the nervous system. Even though the nervous system to a great extent uses chemicals to control the conductance and gain in the neuron, it still is an active device that in some aspects can be modeled by the transistor. A single transistor operated in the subtreshold region doesn't use much more energy than a single neuron (Mead 1990: 1630). Since the nervous system also uses analogue electric signals, Kirchoff's law of current is automatically applicable.

A model of the brain would consist of a description of not only computational primitives, but also the representation of information and organizing principles. We have shown that the computational primitives can be found in analogue electronics.

Carver Mead, who is a well-know scientist within the area of neuromorphic engineering, once said that neural computation is an emergent property of a system, which is only vaguely evident in any single component element (Mead 1989: 5). We actually do know a bit about the inner construction of a neuron, but the true genius of the brain seems to lie in the interconnection of these neurons, the so-called principles of organization.

1.3 ORGANIZING PRINCIPLES

I have not yet mentioned the actual implementation medium of the neuromorphic electronic systems, but that's a neat feature as well: You can use the same VLSI processes as is used for digital circuits, the most popular being CMOS. The building blocks of analogue VLSI (aVLSI) are just the same as the ones for digital VLSI; they are just used in a different way. So even though neuromorphic engineering has not reached commercial viability in but a few areas up to now, we can still build our test-chips for a modest sum of money.

A problem with VLSI though is the mismatch of components, that no two transistors are alike. Since the individual transistors are so small, there is no way the manufacturer can manage to build them exactly like the specification tells them to. In ordinary VLSI, the digital metaphor takes care of the problem, because you only have to make sure that the 1s stay close to V_{dd}[2] and the 0s stay close to G_{nd}[3]. The

[2] Supply voltage.

absolute value of the voltage is not an issue. In analogue circuits we have to find another solution to the problem.

If we study biological "wetware", we can see that the problem actually is much more acute there (Mead 1989:6). Biological systems use adaptive mechanisms to compensate for their lack of precision and their mismatched building blocks. This mechanism also adapts to the dynamics of the building blocks, for like the rest of the body the nervous system will of course go through major changes. Even though new cells develop and old ones die, the system as a whole has to work in much the same way all the time; and therefore the adaptation to the dynamics of the individual building blocks is absolutely necessary. There is also redundancy in the nervous system, so that the system can operate sufficiently even though several neurons stop working.

We would very much like our neuromorphic systems to be as robust and indifferent to absolute device parameters as the nervous system, so redundancy and adaptiveness seems to be the most important organizing principles to implement. Combining redundancy and averaging is one way to implement them both: By having more transistors than is strictly necessary to make the system work, we can take the average over them and so level out the differences. Like the biological adaptive system, this neuromorphic aVLSI counterpart will tolerate faulty active devices. In comparison, a digital system will often stop working as soon as one single transistor breaks down.

Of course there are many more organizing principles inherent in the nervous system; and finding out which, why and how is one of the biggest challenges of neuromorphic engineering.

1.4 REPRESENTATION OF INFORMATION

"Conventional neural networks use large arrays of processing elements, roughly equivalent to neurons; each characterized by an activity level which is often a continuous variable in the range 0 – 1 " (Thorpe et al. 2000: 405). Real neurons instead send a series of all-or-none pulses or spikes. (The signal is discrete in value, but continuous in time.) The neurons integrate their input over time and generate an output pulse/spike when it reaches a certain threshold. This is the "representation of information" component used by our model of the nervous system.

The interconnected neurons communicate by sending such spikes. This a common denominator for most neuromorphic engineering approaches. The most widely accepted form of coding using such spikes is called "rate coding". Varying the average firing frequency of a neuron encodes the information in systems using rate coding. To decode the signal the receiving neuron integrates the signal over time to see how many spikes were fired.

There have always been alternative theories for neural coding, most notably one called temporal coding. Already in 1952 did MacKay & McCulloch show that the pulse trains produced by spiking neurons are much more efficient transmitters of information encoded in relative timings of events rather than numbers of events (Cariani 2001: 737). In temporal coding the information is conveyed by the time of firing of different neurons, not the frequency.

---

[3] Ground.

Empirical studies have shown that a neuron on average only fires a spike every 10ms (Thorpe et al. 2000: 405). If we wanted to code the intensity of a pixel in ten greyscale levels using rate coding it would take up to 100ms to send all the spikes from one neuron. By using 10 neurons in parallel we could do it in 10ms, but the complexity of the circuit would rise proportionally. Recent research has shown that the speed of image processing achieved in the primate visual system is much higher than can be achieved with conventional rate coding (Thorpe et al. 2000: 405).

One solution to keeping the number of neurons low *and* achieve a high speed is called "1atency coding". The time of firing of a simple integrate-and-fire neuron will be proportional to the intensity of the stimulus/input signal. Instead of coding the information as the average firing frequency, as in rate coding, it is encoded as the time of firing of the first spike. (In this way the intensity level is conveyed by one single spike, or the absence of it.) Our earlier example of coding intensity in a greyscale level can now be done with the continuous time scale of firing as opposed to the integration of individual spikes.

Let's say we want to use latency coding on an artificial retina. We connect each pixel to an integrate-and-fire neuron and let the pixel be sensitive to intensity of light. (These "neurons" can be made from the computational primitives of neuromorphic engineering mentioned earlier in this text). The first neuron to fire corresponds to the pixel with the highest intensity, the second neuron to the pixel with the second highest intensity and so on. Pixels with the same intensity will fire at the same time. There are several important advantages to this design:

- The circuit is fast because all happens in parallel and every neuron fires as fast as its threshold is reached.

- There is only one neuron per pixel, so the complexity and energy-usage of the circuit is low.

- Comparing intensity levels in different pixels is very easy. One just has to compare the relative timing of firing from individual neurons. Equality can be checked by a simple AND-gate, since two pixels with the same intensity will fire at the same time and therefore open the AND-gate. Thus the comparison of pixel intensity can be done by asynchronous logic. (This is important for my solution to the visual disparity problem, as I will return to later.)

In 1989 Delbrück and Mahowald built an aVLSI chip which also sought to solve the visual disparity problem. The two solutions are different in almost every aspect of the practical implementation, and attack a slightly different problem. A comparison would require a complete description of the earlier approach. Thus I will concentrate on my own solution here and recommend the interested reader to consult the original paper (Delbrück and Mahowald 1989) on the other.

1.5 SUMMARY

Devoted engineers of digital systems may claim that neuromorphic engineering is a waste of time. The digital circuits are getting faster, smaller, cheaper and more energy-efficient all the time. Surely a DSP-circuit must be able to solve the visual disparity problem? Well yes, but as I have shown; digital technology has its drawbacks like high energy-dissipation, high complexity and little robustness. As

mentioned earlier it is not nearly close to rivaling the performance of the nervous system of even the simplest animals; and working neuromorphic electronic circuits have been produced that are 100 times more efficient in their use of silicon, and 10000 times more energy-efficient than their digital counterparts (Mead 1990: 1636).

The same engineers may claim that at least everything can be *simulated* with digital technology. Well, you *might* be able to simulate it; but the speed will not come close to the actual aVLSI chip. (See chapter 6). Since we hope to build neuromorphic systems that are many times more efficient than digital chips built to solve the same specific problems, simulating these aVLSI chips on generic digital chips will not come close to the real implementation. We will have to design, manufacture and measure actual aVLSI chips to really see what can be accomplished with neuromorphic engineering.

Clearly, neuromorphic engineering may lead to efficient solutions to a number of problems; and we have only started to scratch the surface. The modern desktop computer can solve all the standard computational problems we throw at it; but as soon as you want to rival some aspect of the human nervous system, it falls short. To solve problems the human brain is good at, such as computing visual disparity, neuromorphic engineering with temporal coding seems to be a better technology.

2 VISUAL DISPARITY

When we view a scene there are several monocular cues that allow relative distance and depth to be judged. These include relative size and interposition (Hoey 1998). But it is the binocular cue of visual disparity that allows acute depth discrimination.

As a result of the horizontal displacement of our two eyes they always have slightly different views of a scene. The two images in Figure 1 exemplify this difference. To be able to make a three-dimensional representation of the scene in our mind, the brain tries to pair similar objects in the images and measure their relative disparity. These are called conjugate pairs. Since the offset between the left and right camera is only horizontal, the only possible disparity is horizontal, as shown in Figure 2. Since our eyes have a strictly horizontal displacement, these pairs can only lie along what is called epipolar lines.

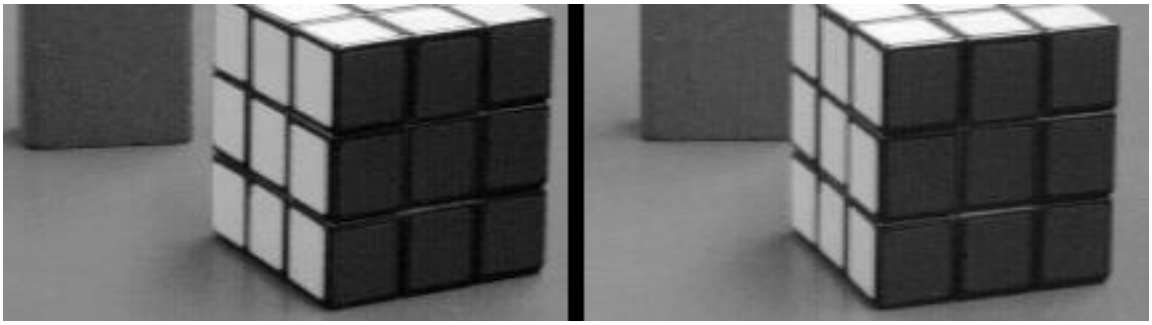


Figure 1 The images of the left and right retinas as a result of visual disparity in stereoscopic vision of a scene. (Lozman et al.1997). (A monocular cue in this figure is the occlusion of one of the cubes in the right image, leading us to believe that the occluded cube is further away. This is called interposition.)

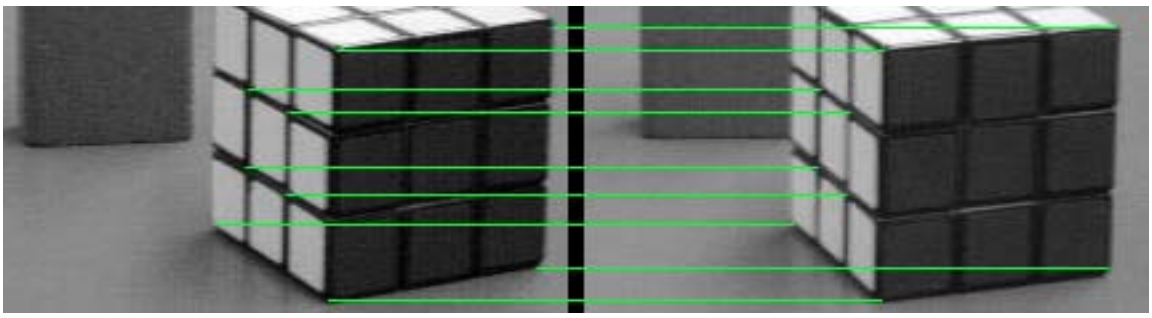


Figure 2 Epipolar lines (Lozman et al. 1997)

The circuit described later assumes input from two one-dimensional sensors, meaning two cameras with pixels along only one horizontal line. The epipolar line restriction is

therefore automatically met since we only compare pixels lying along an epipolar line.

Figure 3 shows all the possible conjugate pairs when one considers 4 retinal projections of objects in each eye. If we assume that there is no shift between the two images, all the right conjugate pairs will lie along the horopter. [4]

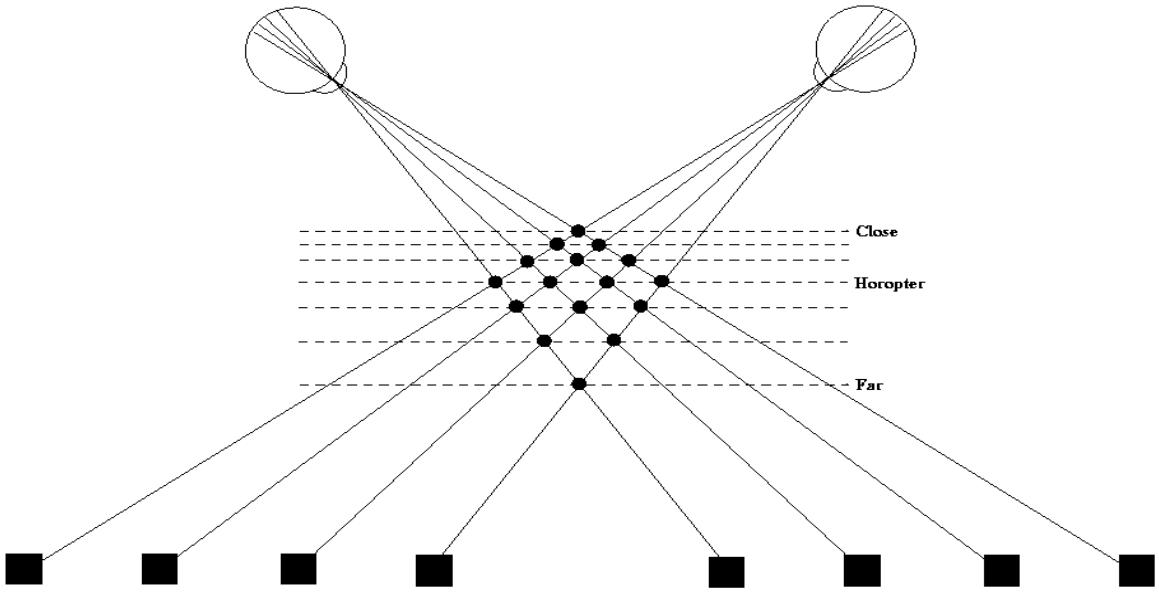


Figure 3 Ambiguity in the retinal projections. It is assumed that the targets (closed squares) correspond to "matchable" descriptive elements obtained from the left and right image. In my implementation these elements are pixels with a continuous greyscale level. Therefore the squares in this figure are all pixels with the same low level of light incident upon them. The lines going through the lens connecting each target/pixel with the retina are lines of sight. The intersections of the lines of sight indicate possible matches between two different pixels. The dotted lines indicate possible disparity planes. In this figure, each of the four targets in one eye's view could match each of the four in the other eye's view. (Match = closed circle.) If we assume that the motive in the images are a plane perpendicular to the angle of sight, only one disparity can be right. (Of the 16 possible matches only 4 can be right, the other 12 wrong.) If there is no shift between the two images, the horopter is the right disparity plane. (Redrawn from Marr and Poggio 1976: 285).

The example in figure 4 with four conjugate pairs has only *one* possibility of visual disparity, since all these pairs must lie along the same plane.

Using pixels as the objects to compare from each image does not fulfill the uniqueness constraint (Marr and Poggio 1976: 284), which states that each item from each image may be assigned at most one disparity level. This means that one has to match objects that are unique in each image to find a conjugate pair. A practical implementation would for instance use edge detection to single out comparable objects in the two images.

[4] The horopter is the locus of points in space that stimulate corresponding retinal points. One could also say that it is the location of objects in space that give rise to zero retinal disparity.

In my implementation I have assumed that the scene viewed by the two cameras is a plane perpendicular to the angle of sight[5]. This leads to a practical solution to the problem of not fulfilling the uniqueness constraint: In the image of the plane, all the pixels must have the same visual disparity. In figure 4 that means that all the closed circles have to lie along *one* dotted line. Since there is only one possible match for each of the four pixels in figure 4, they all lie along one of these lines.

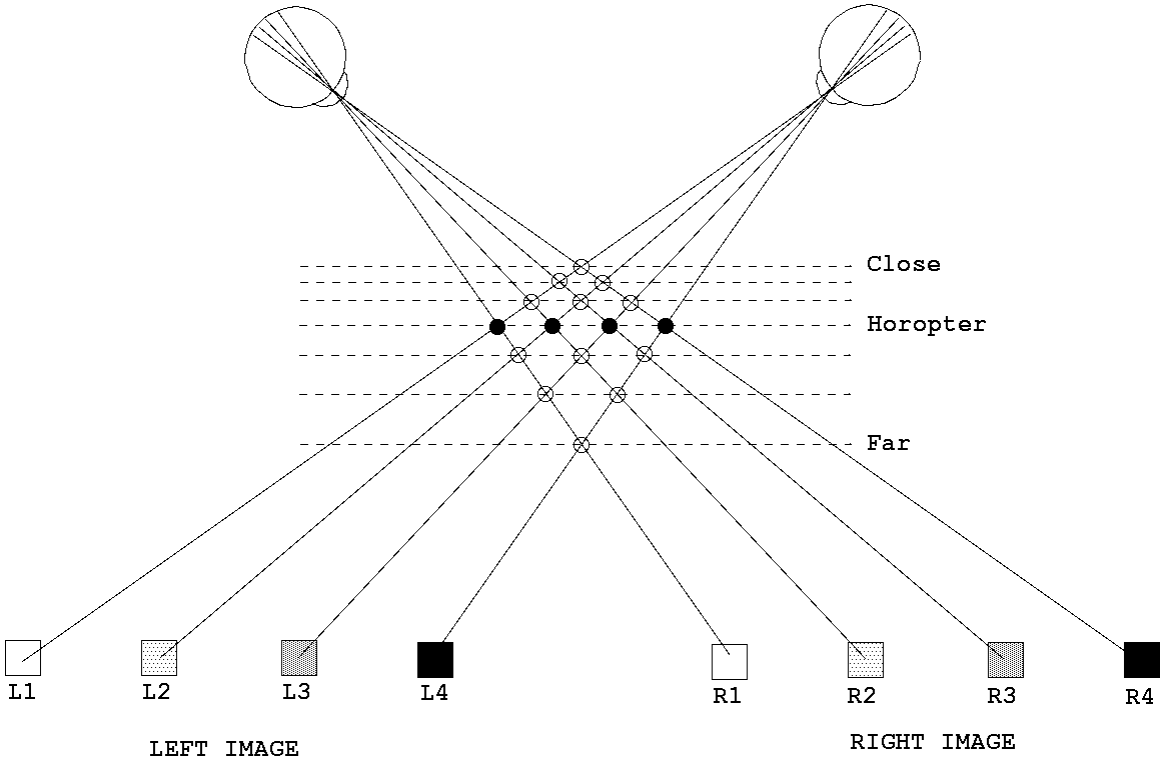


Figure 4 As opposed to the situation in figure 3; here we have no ambiguity in the retinal projections. In this figure there is only 4 conjugate pairs because there are 4 different greyscale levels in each image. Thus each pixel in one image only matches one of the pixels in the other image. (Match = closed circle. No match = open circle.)

Figure 5 shows an example of having several disparities for some of the objects/pixels. But since the disparity of all pixels should be the same one can assume that the disparity most frequently encountered is the right one.

[5] I also assume that the lighting source is a point source at infinity, the surface is lambertian and that the amount of figural dissimilarity or distortion between the views is small. (Lane and Thacker 1996)

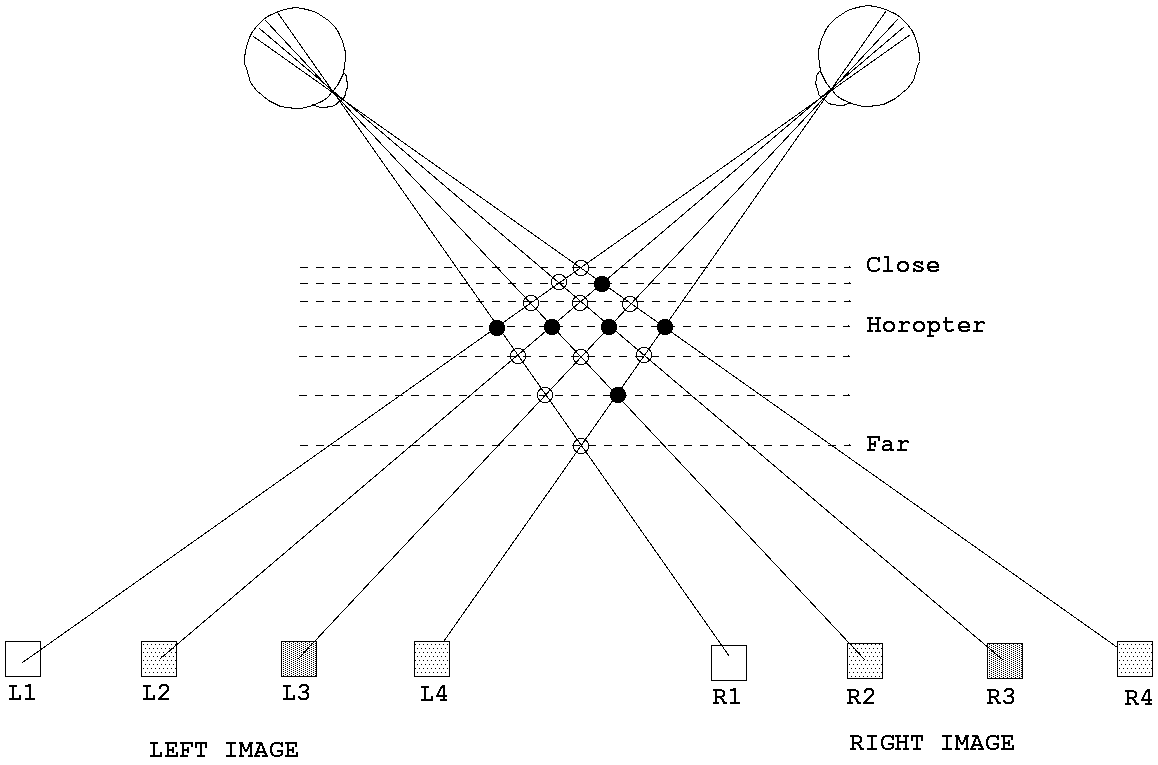


Figure 5 Lack of uniqueness in the pixels compared gives false matches. Since the images are not shifted in relation to each other, all the right matches lie along the horopter. (Match = closed circle. No match = open circle).

2.1 A FIRST LOOK AT A PRACTICAL IMPLEMENTATION

My implementation's main principle of operation can be explained by a new look at figure 5. A correlator with two inputs replaces each circle. Each correlator's input is the two pixels given by the intersection of lines of sight. If the two pixels match, a spike is sent along the horizontal dotted line it is placed upon. (The horizontal dotted line represents disparity planes.) If they don't match, no spike is sent. The number of spikes along each line is summed up and the sum of each line is compared. The line with the highest number of spikes represents the right visual disparity. In figure 5 this would mean that the horopter is chosen as the winner (, with the number 4.) In figure 6 the horopter wins again (, with the number 5.) In the latter figure, the wrong disparity is found. Or actually the horopter *could* be the right answer, but so can several other disparities. The problem is that the number of correlators on the horopter is higher than on any other line, so it will win even though it might not be the right winner. This is also the case for several other input images and must be rectified, or else it will be very difficult for the lines with the least correlators to ever win.

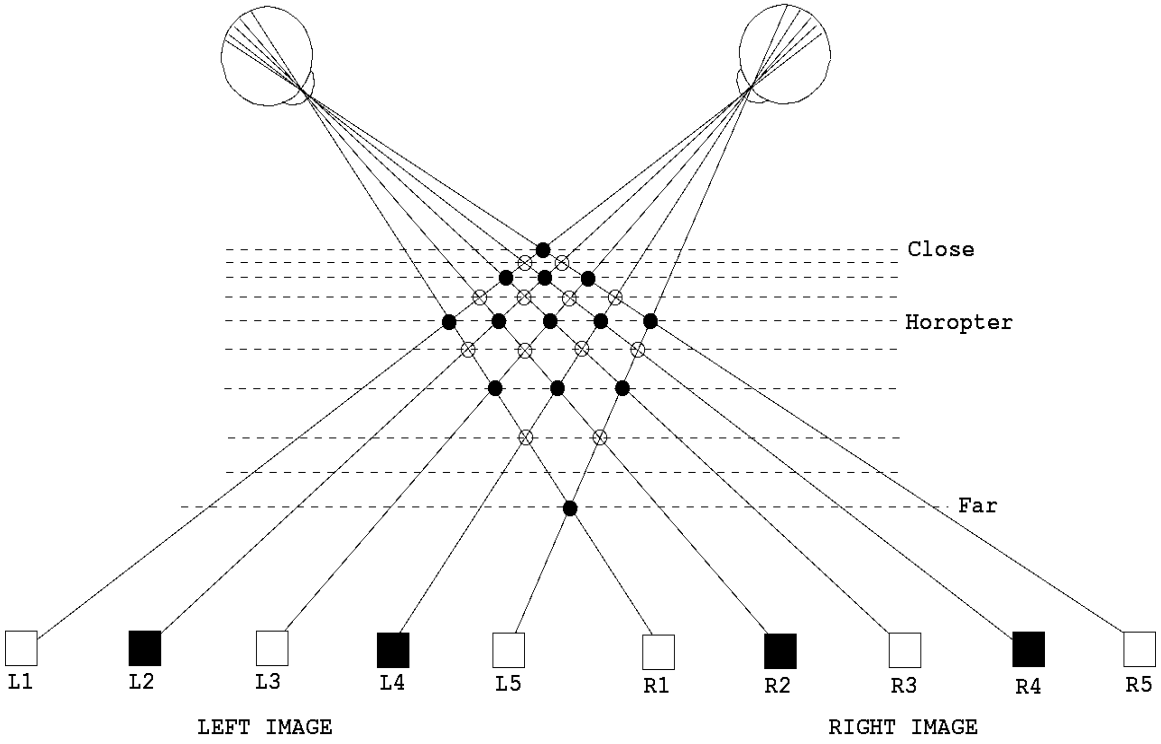


Figure 6 An image of a pattern give rise to ambiguity. There are several possibilities for matching one object in the left eyes view to objects in the right eyes view. (Match = closed circle. No match = open circle). There is no way of telling what the right disparity is.

The solution is to put the same number of correlators on each line. By doing this none of the lines have an artificial advantage, they all have the same possibility of becoming the winner. Figure 7 shows this solution with the same input as in figure 6, i.e. a pattern. Clearly none of the possible disparities becomes the winner, their sums are the same. This reflects the fact that we cannot know what the right disparity is.

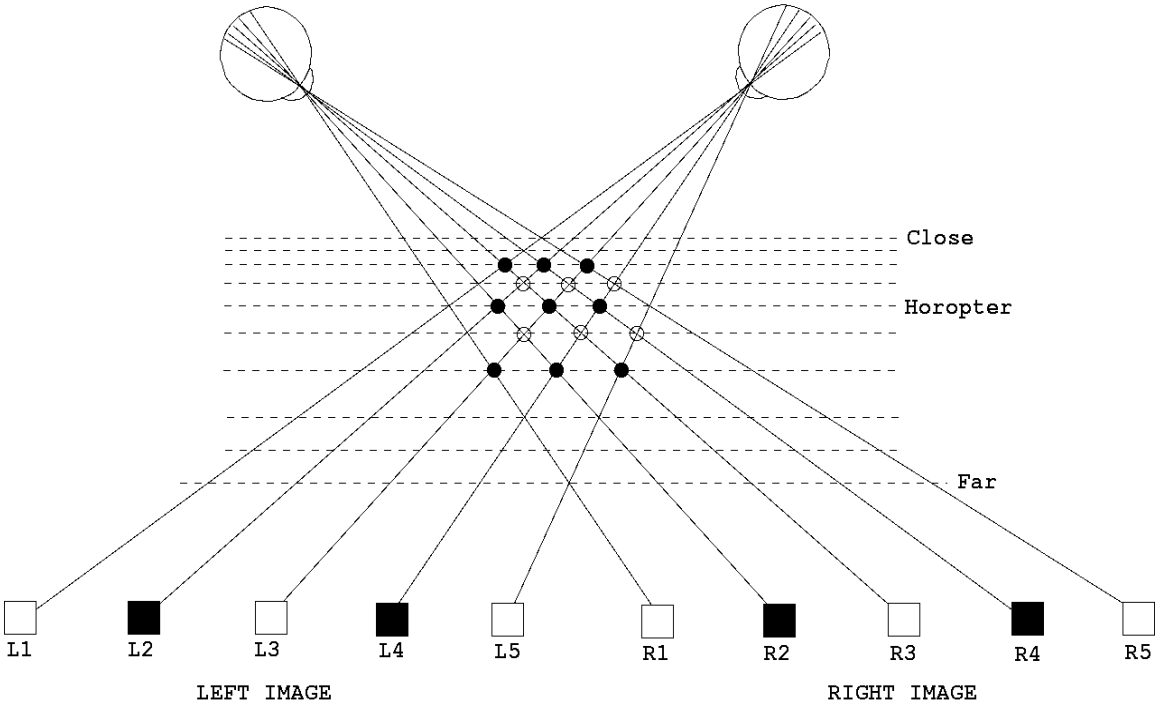


Figure 7 An image of a pattern give rise to ambiguity. There are several possibilities for matching one object in the left eyes view to objects in the right eyes view. (Match = closed circle. No match = open circle). There is no way of telling what the right disparity is. This can be seen as the same number of matches on three of the horizontal lines/disparity planes.

3 MATERIALS AND METHODS

The biological systems that neuromorphic engineering draws its inspiration from are all developed through evolution. It is this very simple, but yet incredibly powerful tool that has made the vast amount of biological systems so superior to the manmade imitations. Therefore it has been natural for me to approach the task of solving the problem of computing visual disparity in the same way.

I set out to do this with the idea of harnessing the power of analogue computation and the idea that comparing intensity of pixels using temporal codes is less complex and more efficient than by rate coding. The rest has been a combination of knowledge, and trial and error.

Planning a large analogue system, like the one presented in this thesis, is hard to do without trial and error. There are very many parameters to assess; and doing it all in one go, without frequent simulations, is very tough. Therefore I split the problem into small manageable parts, and used the following evolutionary process to solve them:

1. Provide a mutation in the form of a new idea to solve a specific problem or design a particular part of the system.

2. Do a software simulation to provide the tool for "natural selection".

3. Use my supervisors, my colleagues and myself as an instrument for weeding out the solutions that weren't "fit".

First I used this method to develop the individual circuits presented in chapter 5. Then I used it to develop the compound circuits seen as the individual blocks of the Figure 14. Then I developed the total system on schematic level using the same method. Some minor problems led me to fine tune the individual circuits presented in chapter 5. When the total system on schematic level showed a satisfactory performance, as shown in chapter 6, I designed a matching layout. There are a number of problems inherent in the real world of VLSI chips, not least process variations, which is hard to simulate on computer software. So the ultimate test/judge would have to be real experiments on a physical chip.

Cadence, from Cadence Design Systems, was the software that provided the whole package of CAD tools; from the design of schematics and analysis of these, to the layout and verification of this towards the schematic. The chip was to be produced by Austria Micro Systems in an $0.6\mu m$ mixed-signal process, so I used their corresponding hit-kit to provide the right process parameters for the Cadence software.

The final experiments on the physical chip were performed using the Matlab mathematical computation software package from The Mathworks INC; and a HP16500

Logic Analyzer. The software was run on a SUN workstation with the Solaris operating system, which had a network connection to the HP.

I used the software to synthetically produce sensor stimuli for the chip. I did this partly because I didn't have access to two suitable cameras that could work as the left and right sensor for computing visual disparity. The advantage of using synthetically produced stimuli was a more controlled test environment than would be the case with two cameras. Knowing the exact specification of the stimuli made it easy to be sure what the output of the chip due to the input should be. The stimuli were in the form of a list of AER events.

The files containing stimuli where transferred to the HP via the network connection. The HP had a parallel 7-bit connection to the AER on-chip receiver, plus 2 bits for standard acknowledge and receive signaling. It had a similar connection to the AER on-chip sender to store the chip output due to the some input stimuli. The output was stored in the form of a list of AER events similar to the stimuli file. This file was transferred to the Sun workstation for analysis in the Matlab software. The figures presented in chapter 8 were produced using this software.

4 IMPLEMENTATION ON FLOW CHART LEVEL

Since the system we want to build should use spikes for the representation of information, lets assume that each of the pixels in the two cameras is connected to an integrate-and-fire-neuron. If the pixel has an output current proportional to the intensity of light falling upon it, the intensity will be coded in the average frequency of firing of the neuron. Using this form of rate coding will require quite complicated correlators. One could for instance integrate the spikes on the two input channels and use the voltage on the capacitances to drive a transconductance amplifier. This would give a large output current for dissimilar pixels and none for similar pixels.

In my implementation I use two "cameras" with 128 pixels each. This makes it possible to have 65 different disparity planes by using 64 correlators on each plane. This adds up to 4160 correlators, and therefore each one of them should be quite simple to avoid making the circuit to complex. (By "simple" I mean having few transistors). By using temporal coding the correlator can be a simple AND-gate. (I will from this point on refer to the correlators as coincidence detectors, since a coincidence in time between to spikes corresponds to a correlation between the sizes of two voltages or currents.) Since the intensity of light falling on a pixel is encoded in the latency of firing, the coincidence detector just checks if the timing of the spike sent out from two pixels is the same. If the timing is the same the intensity of light is also the same. One also has the advantage of speed, as the coincidence detectors don't have to integrate the input over time. To use this temporal coding scheme one has to have one-shot integrate-and-fire-neurons connected to the pixels. "One-shot" means that they will only fire once before being manually reset. If one were to use ordinary integrate-and-fire neurons the pixels with high intensity would have a higher frequency of firing and lead to high numbers of coincidence events dominating the summing on each disparity line.

One also has to have a common time reference for all pixels when using latency coding. So the one-shot neurons have a manual reset driven by a clock signal.

During the following explanation of the main principles of operation I assume a fictitious system with four pixels in each image as shown in fig.8.

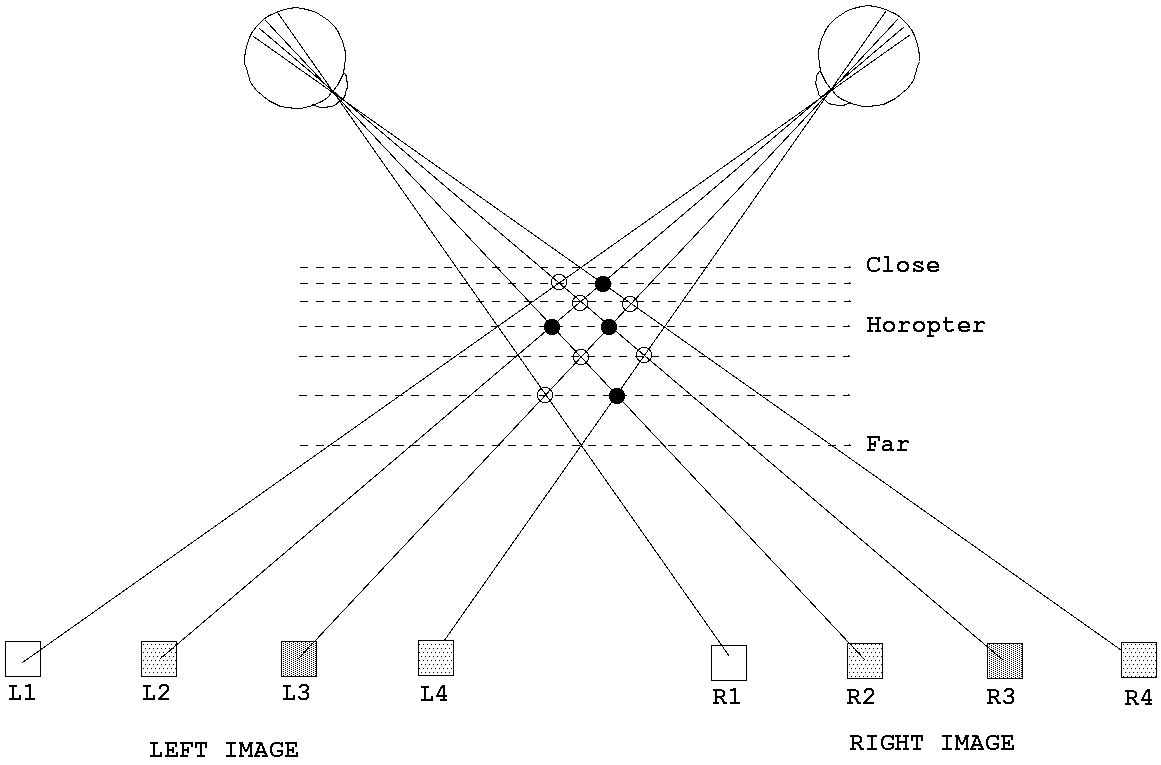


Figure 8 Image of pixels with random intensities. (Match = closed circle. No match = open circle.)

These images are in reality 8 channels of current fed to 8 separate input neurons. These neurons convert the continuous current representation of the visual sensor to a spike representation fed to the coincidence matrix (, see Figure 9). The latency of firing is inversely proportional to the size of the current.

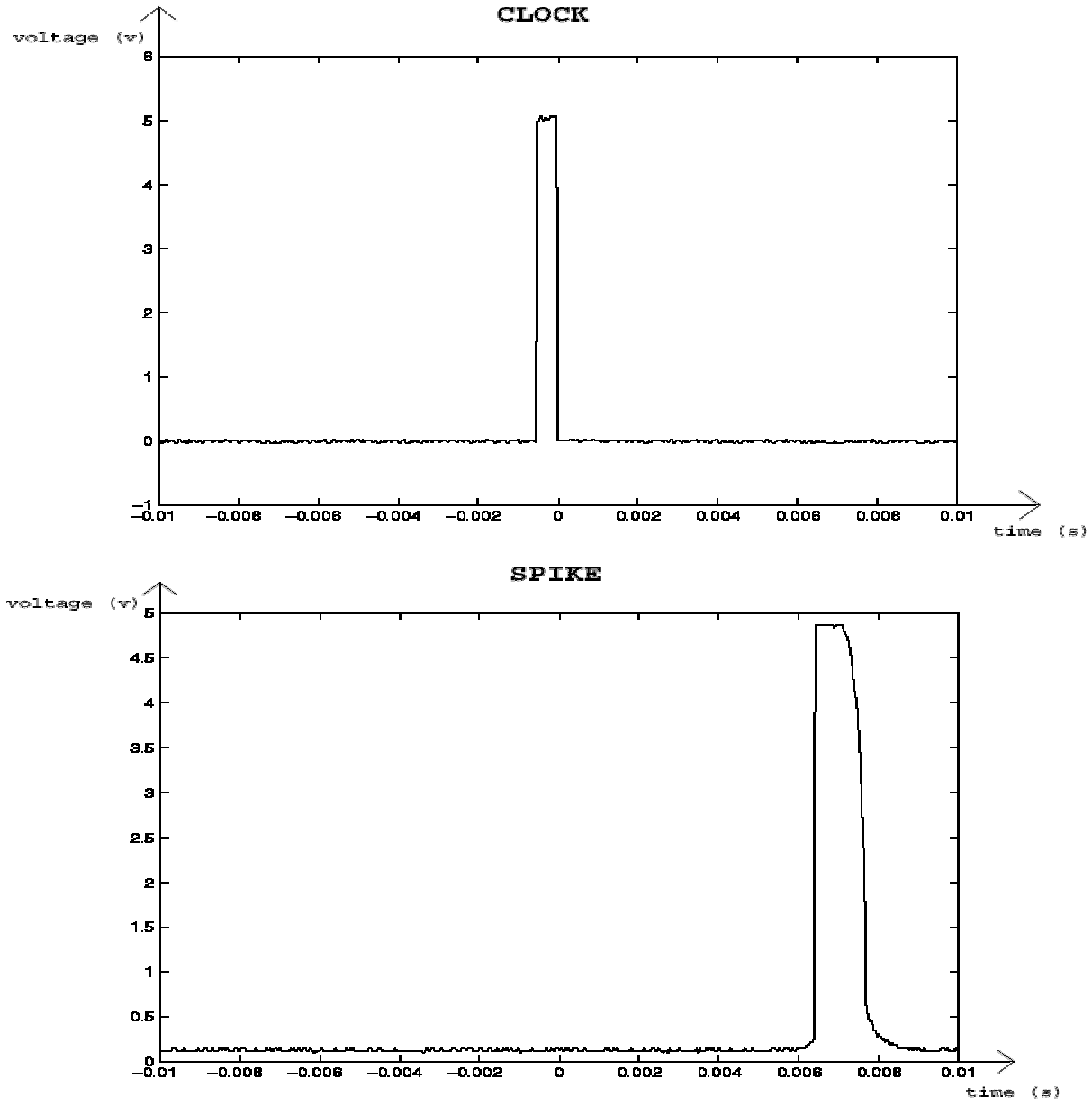


Figure 9 Response of one single input neuron on the final VLSI chip (lower part) with the clock as source of reset (upper part). Clock frequency is 100Hz. Larger current input would make the spike come closer to the falling flank of the clock signal, and vice versa. (The reset is active high. This snapshot is taken from an oscilloscope during the final experiments on the actual chip).

Figure 10 gives a simple overview of the circuit from stimuli to the coincidence matrix.

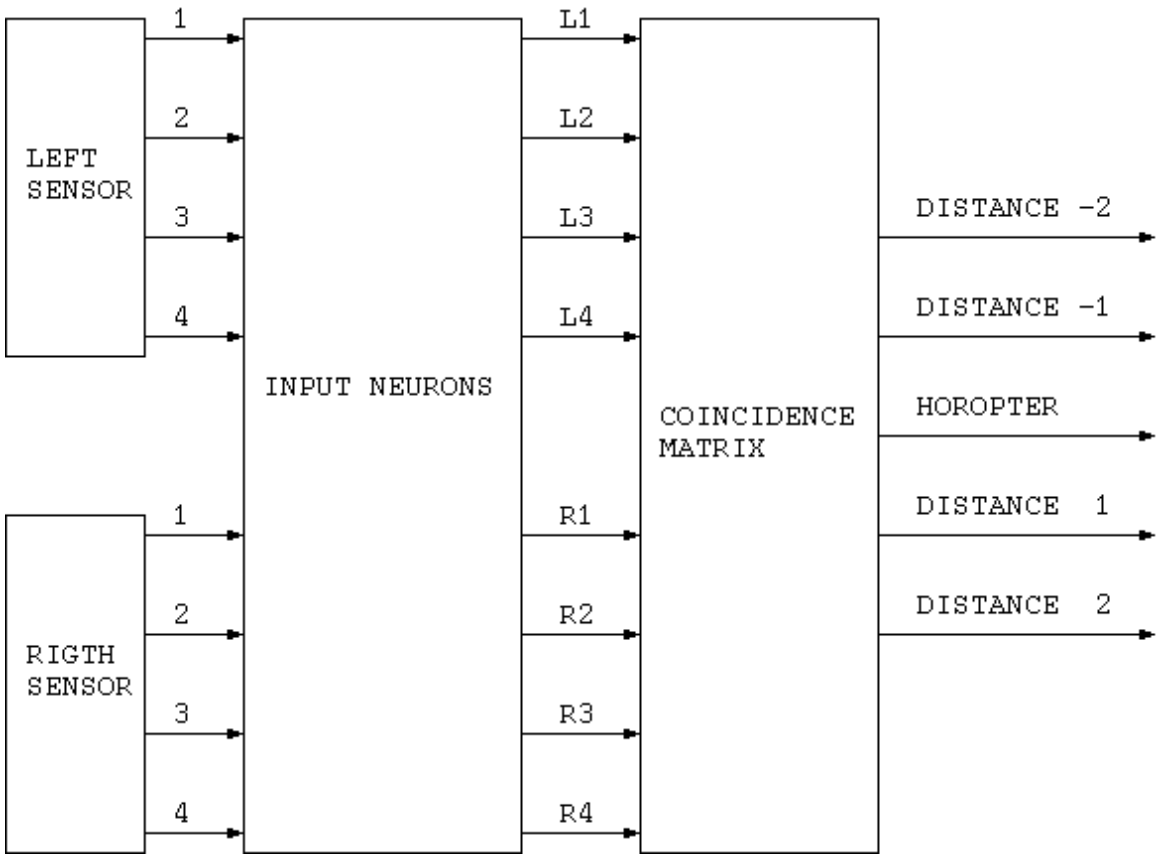


Figure 10 Simplified flowchart of the first three stages of the stereopsis circuit. (The left and right sensors are not implemented in the final VLSI chip. It uses synthetic test images provided through a 7-bit AER communications system (APPENDIX A: AER). The sensors are included here for ease of understanding). There is one separate input neuron for each pixel. These transform current into spikes. The spikes are then sent to a matrix of coincidence detectors as shown in Figure 13. This matrix implements the connections as shown in Figure 8.

The amount of current flowing from a coincidence detector due to an overlap in time between the spikes on its two inputs is not absolute. (This was a simplification done in chapter 2.1.) Since the coincidence detector is a simple AND-gate there will be current flowing as long as the two spikes are high. The total charge due to a coincidence will depend upon the degree of overlap in time. A simulation of different levels of coincidence, and the resulting current output, can be seen in Figure 11 and Figure 12.

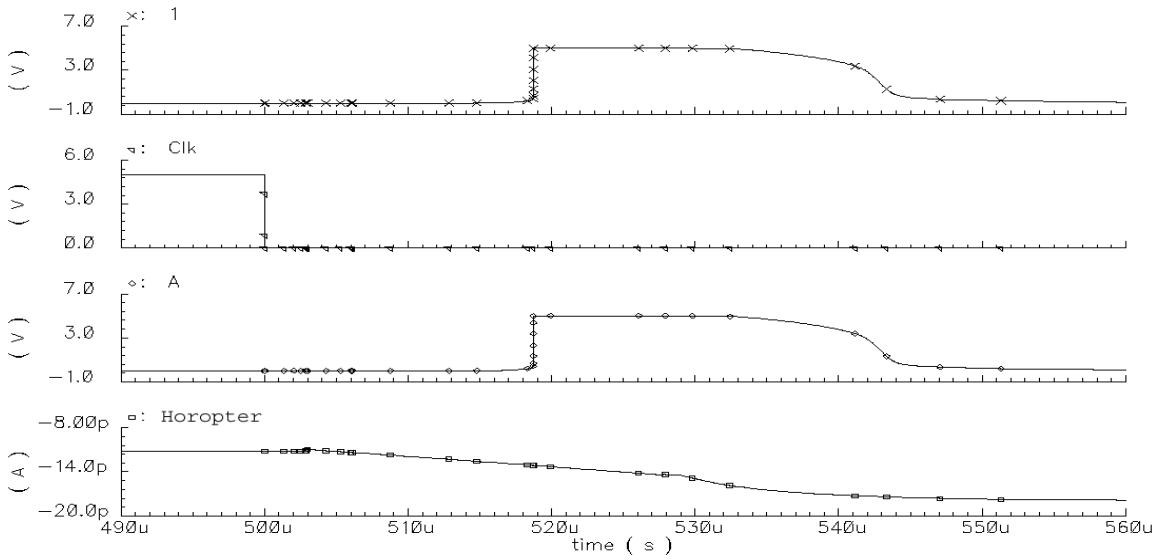


Figure 11 Spike overlap and its effect on the total current on the corresponding disparity plane. Pixel 1 and pixel A are connected to the same coincidence detector on the horopter. There are several other coincidence events on the same disparity plane since the horopter is the right answer in this simulation. Therefore the current steadily rises. (We use a sinking of current, therefore the negative sign.)

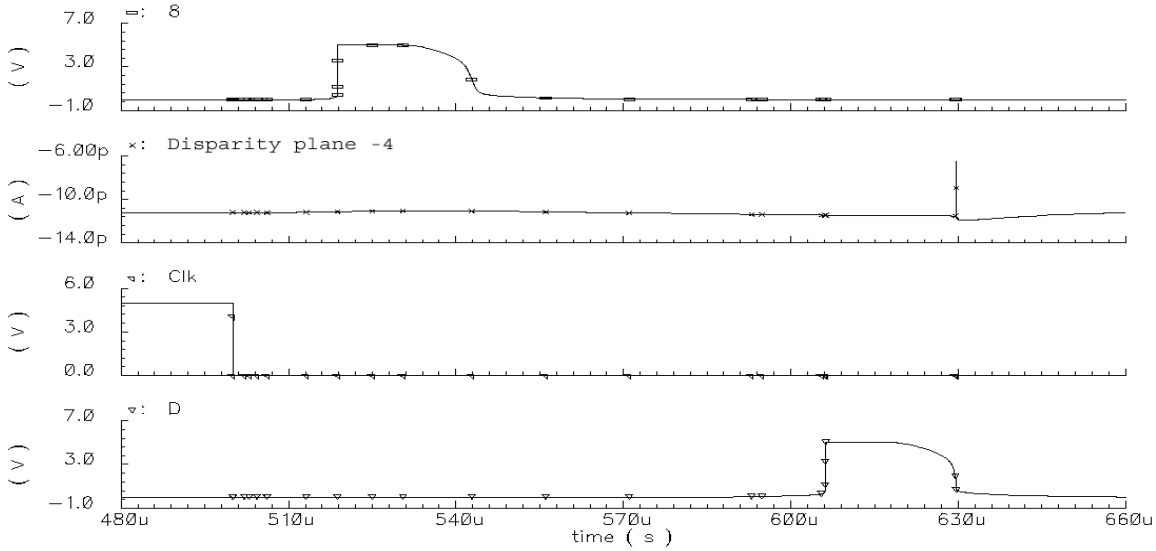


Figure 12 Lack of spike overlap and the corresponding lack of effect on the total current on the corresponding disparity plane. Pixel 8 and pixel D are connected to the same coincidence detector on disparity plane -4. This plane does not represent the right disparity, and so the current is quite stable throughout the period. The small peak on disparity plane -4 is noise. (The plot is taken from the same simulation as the one in Figure 11.).

We need to integrate the current on each disparity plane of the matrix to sum the number of coincident spikes in each period. We do this by adding a capacitance on each plane. (It is placed in the current mirrors of Figure 14.)

At all times there is only one true disparity if we assume that the scene viewed by the two cameras is a plane perpendicular to the angle of sight. By adding a winner-take-all (WTA) system to the circuit we implement this constraint. The WTA will at all times suppress all the weakest channels, and only relay the strongest one; i.e. the one with the highest current.

Figure 13 shows the main connection scheme of the coincidence matrix.

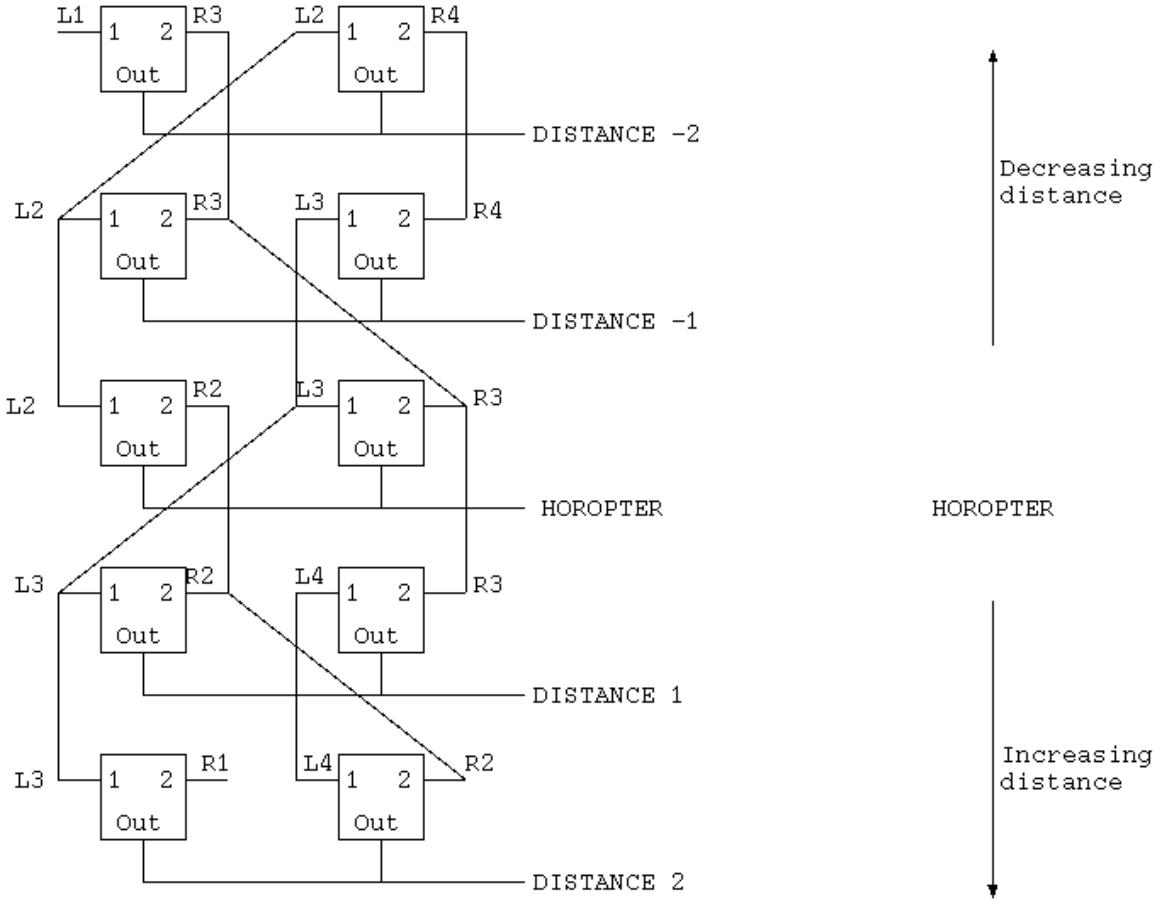


Figure 13 Coincidence matrix. Each of the squares represents a coincidence detector with two inputs and one output. The matrix implements the connections shown in Figure 8. Each of the letter/number combinations represents an input from the corresponding neurons. So R3 is the output of the integrate-and-fire neuron connected to pixel number 3 in the right sensor/camera.

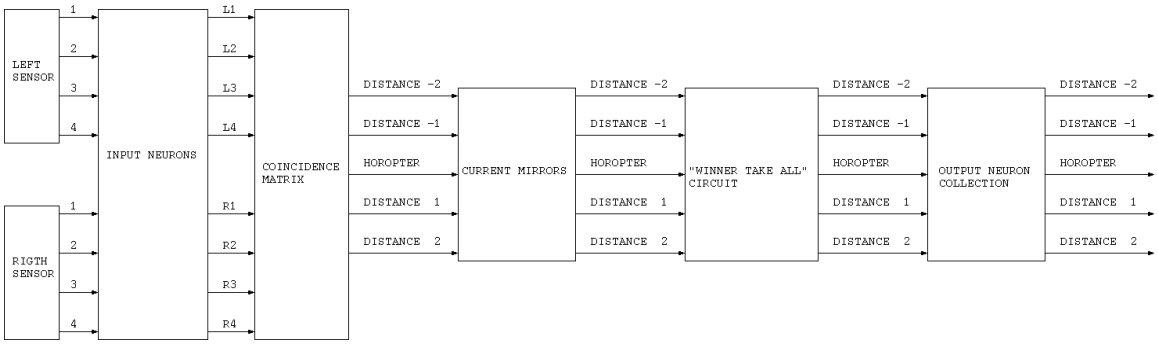


Figure 14 Simplified flowchart of the final implementation. The left and right sensors of this figure are added for simplicity of understanding. In reality these stimuli come from an AER receiver. The AER sender located after the output neurons is also omitted.

After the WTA of Figure 14 we have simple integrate-and-fire neurons to convert the current output of the WTA to spikes suitable for off-chip communication through the AER communication standard. (The AER sender is not shown in the figure).

5 IMPLEMENTATION ON CIRCUIT LEVEL

5.1 INPUT NEURON

The one-shot integrate-and-fire neuron is built around a simple RS-latch. The latch is a standard design using two cross-connected NOR-gates, but unlike textbook RS-latches this one doesn't have an undefined state when both R and S inputs are high. (For reference the truth table of this RS-latch is given in Table 1.) The simulation of the RS-latch is shown in Figure 15.

Table 1 Truth table for my specific implementation of an RS-latch. The thing that sets it apart from textbook cases is the fact that having both R and S inputs high is not an undefined state. This situation will always lead to both a low Q and –Q output, as can be seen in the simulation of Figure 15.

R	***S***	***Q***	***-Q***	***Comment***
0	0	Q	-Q	Holds the last value.
0	1	1	0	Set
1	0	0	1	Reset
1	1	0	0	Usually undefined state.

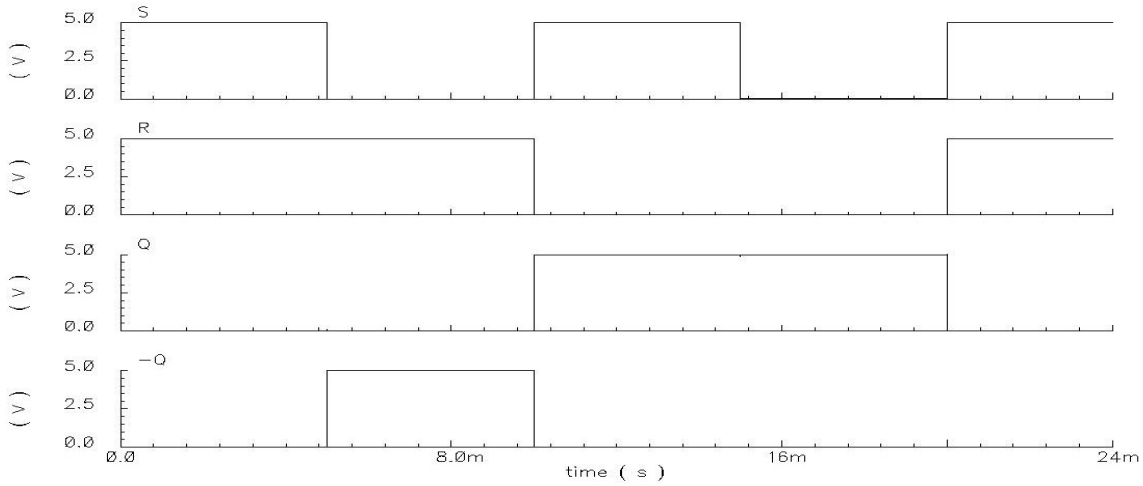


Figure 15 Simulation of the RS-latch. As can be seen it is stable and well defined also when both input R and S are high.

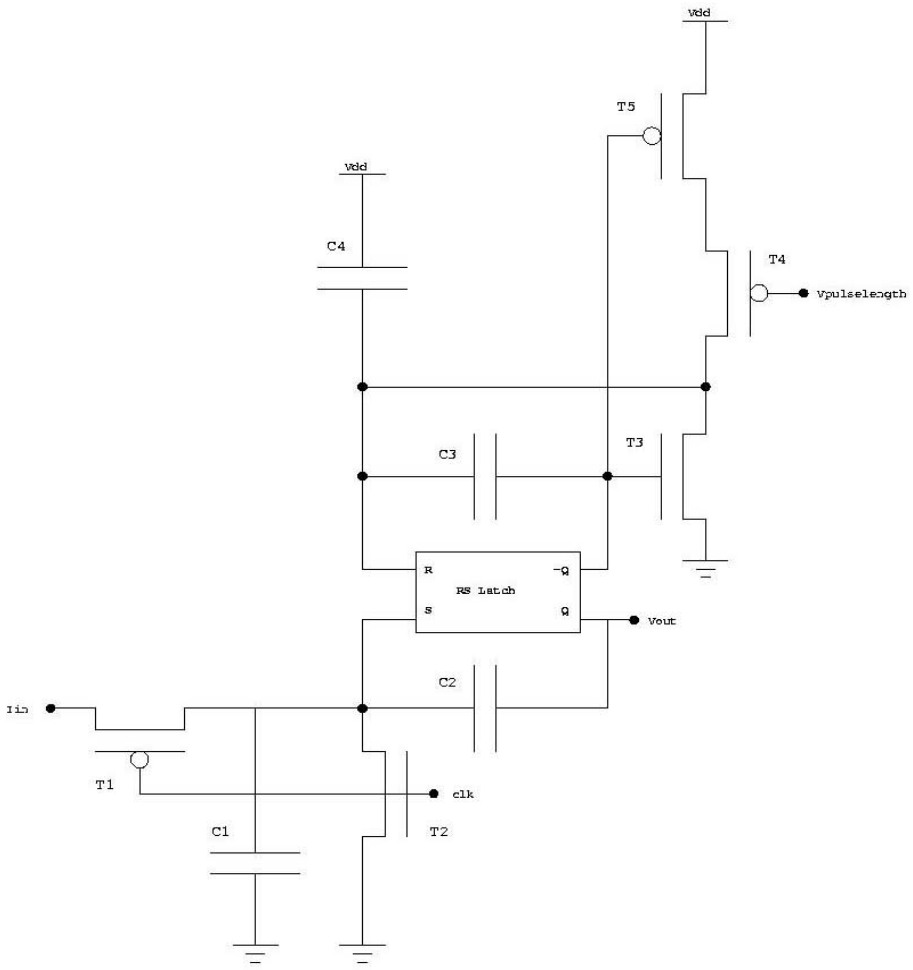


Figure 16 One-shot integrate-and-fire neuron. The neuron will integrate the current on input Iin and fire a spike on output Vout when a threshold is reached. It cannot fire another spike until it is manually reset by a high clock signal (clk). Vpulselength is a bias to control the length of the spike issued. By lowering the voltage the spike gets shorter. (Designed by Philipp Häfliger of the Microelectronic Systems Group, Dept. of Informatics, University of Oslo)

The capacitance C1 (Figure 16) will integrate the current Iin through the T1 transistor while the clock (clk in Figure 16) is low. When the voltage on C1 (, the same as S in Figure 15,) reaches 1.9V the RS-latch will start to switch. The voltage on Q rises, and it is fed back through capacitance C2 to pull S towards Vdd. This makes the RS-latch switch very fast; and Q, which is also the output (, Vout in Figure 16), goes high almost instantly (Figure 17). At the same time -Q goes low and this starts charging capacitance C4 through the T4 and T5 transistors. When the voltage on input R reaches 1.9V the RS-latch will make both Q and -Q go low (see Figure 17.) This ends the spike on Vout, but both S and R remain high. Thus the neuron cannot fire again until S goes low and we have a proper reset. When the clock goes high the charge on C1 is drained through transistor T2. Thus the flow of input current is

temporarily cut off with transistor T1. Since S now goes low but R stays high the RS-latch is reset. This opens T3 so the charge on C4 can drain out through T3. (C3 is added to avoid oscillations in this stage of operation.) Now R and S is both low and the neuron is ready to fire again as soon as the clock goes low. So this is a one-shot integrate-and-fire neuron that can only fire once every clock cycle. The fact that the specific RS-latch we use has a well-defined output when both R and S are high is a crucial fact to make this neuron work.

Vpulselength (Figure 16) is a bias to control the length of the spike issued. A lower voltage will increase the current through PMOS transistors T5 and T4 when –Q is low. This will reduce the time it takes to charge capacitance C3 to 1.9V, thereby reducing the length of the spike. A shorter spike will give a smaller time window of coincidence, and this will lower the tolerance for pixel similarity.

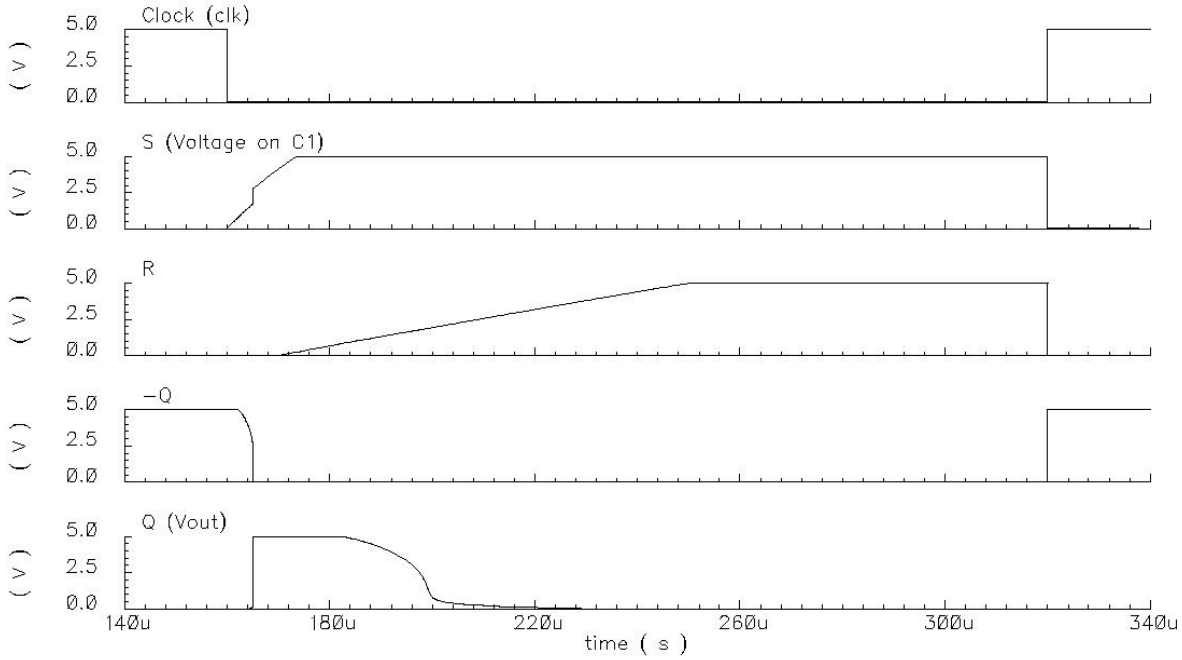


Figure 17 Simulation of the one-shot integrate-and-fire neuron shown in Figure 16. We feed the neuron with a suitable, steady current input. The input of the neuron is closed as long as the clock is high, but as soon as it goes low the neuron starts integrating the current on capacitance C1. When S, which is the voltage on the capacitance, reaches 1.9V the output Q goes high. This change is fed back through capacitance C2 and pulls S towards Vdd (=5V). At the same time the voltage R *also rises, due to –Q going low, and when it reaches 1.9V the output Q goes low again. The time it takes* R *to reach 1.9V is controlled by the bias Vpulselength. Since both S and* R *stay high the neuron cannot fire until the clock has reset the circuit.*

5.2 COINCIDENCE DETECTOR

I have described the principal construction of the coincidence detector in chapter 4. The only modifications to the description given there is the addition of an NMOS transistor and the fact that we use a NAND gate instead of an AND gate, see Figure 18.

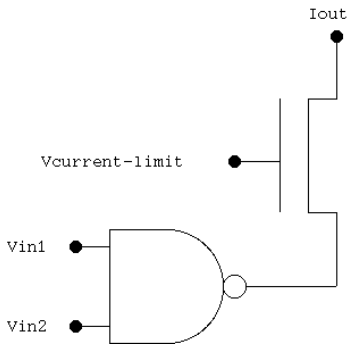


Figure 18 Coincidence detector. By using temporal codes it can be made extremely simple, as this one. (We use a NAND gate because we want to sink current when there is a coincidence in time between the two input spikes.) Vcurrent-limit is a bias to control the amount of current due to a coincidence. By adjusting it we control the amount of smearing/integration in the capacitance of the following current-mirror (Figure 19).

The NMOS transistor is added to be able to control the amount of current flowing during a coincidence event. The element following the coincidence detectors of the coincidence matrix is a current mirror with a capacitance added, as described in chapter 5.3. This capacitance is used to sum the coincidence events of each disparity line. The right disparity is the line with the highest number of coincidence events. Each event should not charge the capacitance very much, because this would lead to a quite unstable output of the current mirror. This in turn would lead to an unstable winner-take-all circuit that changes rapidly between different winners.

Since the medium of implementation is VLSI and the area is sparse, the size of each capacitance used is very small. (The value of capacitance is only about 160fF.) So we have to set the bias Vcurrent-limit low so the NMOS transistor operates in the subtreshold region of operation. This is also important to keep the energy dissipation of the circuit low. There are 65 disparity planes of 32 coincidence detectors in the final implementation. This adds up to a total of 2080 coincidence detectors. A totally uniform picture would actually lead to a simultaneous coincidence event in all the coincidence detectors at the exact same time. (The intensity of the pixels must be high enough to make the input neurons fire during clock low.) This will give large energy dissipation during a short period of time; therefore we have to keep the current due to a single coincidence event low to keep the total energy dissipation of the chip low.

The simplicity of the coincidence detector is also a large size benefit. The area of the chip covered by the coincidence matrix in the final implementation is about 50% of the total area covered by my circuit, not including the AER communication circuitry. The number of coincidence detectors can be found by the following formula:

$$Number\_of\_coincidence\_detectors = \frac{Pixels}{2} \times (Pixels + 1) \approx \frac{Pixels^2}{2}$$ [6]

The number of output and input neurons, current mirrors and WTA parts can be found by using this formula:

$$Number\_of\_parts = Pixels + 1 \approx Pixels$$ [6]

As the two above formulas show the number of coincidence detectors grows quadraticaly, while the number of other parts only grows linearly, as we incorporate more pixels in each sensor to improve the precision of the system. So it is of utmost importance to keep the size of the coincidence detector as small as possible. As mentioned earlier the use of temporal coding facilitates the use of such simple coincidence detectors. (See Figure 18.)

5.3 CURRENT MIRROR

In the previous chapter I explained the reason for having a capacitance in the current mirror. The current mirror uses the voltage on this capacitance to give a stable current to the following WTA. In addition to having added a capacitance, we have also opted for having separate power supplies for the left and right PMOS transistor; see Figure 19. By adjusting the ratio between the two supply voltages we can adjust the amount of amplification. Vdd right/Vdd left > 1 will give a positive amplification, and Vdd right/Vdd left < 1 will give a negative amplification. We tweak this amplification to obtain a suitable input for the WTA system.

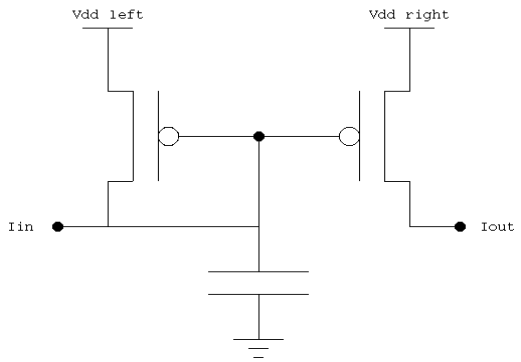


Figure 19 A standard current mirror with a capacitance added for summing the spikes on each disparity plane of the coincidence matrix. There are separate power supplies for the left and right transistor. By adjusting the ratio between them we can adjust the amplification of the current mirror.

(An in-depth explanation of a general current-mirror can be found in (Mead 1989: 39-40). It is not given here since a current-mirror is a standard electronic circuit.)

[6] "Pixels" are the number of pixels in each of the two sensors used to provide input for the visual disparity-computing chip.

5.4 WINNER-TAKE-ALL (WTA) CIRCUIT

The WTA system used here is basically the same as the one designed by Lazzaro, Ryckebusch, Mahowald and Mead of the California Institute of Technology. I refer to their paper "Winner-take-all networks of O(n) complexity" (Lazzaro et al. 1988) for an in-depth explanation of the functionality of the WTA system. The basic explanation they give is the following:

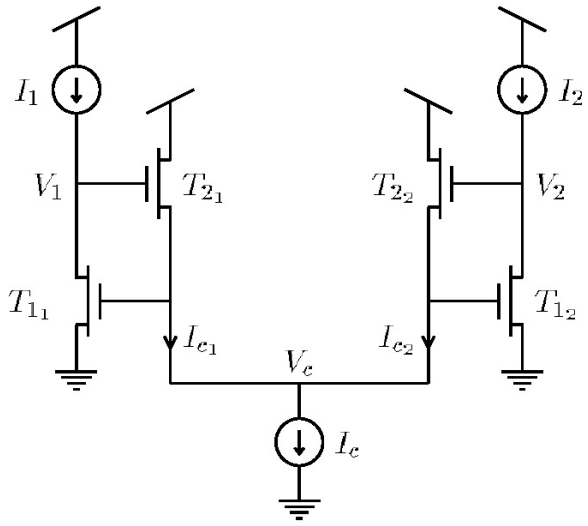


Figure 20 Schematic diagram of a two-neuron winner-take-all circuit

The above figure shows a schematic diagram of a two-neuron winner-take-all circuit. To understand the behavior of the circuit, we first consider the input condition $I_1 = I_2 \equiv I_m$. *Transistors* T_{1_1} *and* T_{1_2} *have identical potentials at gate and source, and are both sinking* I_m*; thus, the drain potentials* V_1 *and* V_2 *must be equal. Transistors* T_{2_1} *and* T_{2_2} *have identical source, drain, and gate potentials, and therefore must sink the identical current* $I_{C_1} = I_{C_2} = I_C / 2$. *In the subtreshold region of operation, the equation* $I_m = I_o \exp(V_c / V_o)$ *describes transistors* T_{1_1} *and* T_{1_2}, *where* I_o *is a fabrication parameter, and* $V_o = kT / qk$. *Likewise, the equation* $I_c / 2 = I_o \exp((V_m - V_c) / V_o)$, *where* $V_m \equiv V_1 = V_2$, *describes transistors* T_{2_1} *and* T_{2_2}. *Solving for* $\mathrm{V_m(I_m, I_c)}$ *yields*

$$\mathrm{V_m = V_o\ ln(\frac{I_m}{I_o}) + V_o\ ln(\frac{I_c}{2I_o})}.$$

Thus, for equal input currents, the circuit produces equal output voltages; this behavior is desirable for a winner-take-all circuit. In addition, the output voltage V_m logarithmically encodes the magnitude of the input current I_m.
The input condition $I_1 = I_m + \delta_i$, $I_2=I_m$, illustrates the inhibitory action of the circuit. Transistor T_{1_1} must sink δ_i more current than in the previous example; as a result, the gate voltage of T_{1_1} rises. Transistors T_{1_1} and T_{1_2} share a common gate, however; thus, T_{1_2} must also sink $I_m + \delta_i$. But only I_m is present at the drain of T_{1_2}. To compensate, the drain voltage of T_{1_2}, V_2, must decrease. For small $\delta_i s$, the Early effect serves to decrease the current through T_{1_2}, decreasing V_2 linearly with δ_i. For large $\delta_i s$, T_{1_2} must leave saturation, driving V_2 to approximately 0 volts. As desired, the output associated with the smaller input diminishes. For large $\delta_i s$, $I_{c_2} \approx 0$ and $I_{c_1} \approx I_c$. The equation $I_m + \delta_i = I_o \exp(V_c / V_o)$ describes transistor T_{1_1}, and the equation $I_c = I_o \exp((V_1 - V_c)/V_o)$ describes transistor T_{2_1}. Solving for V_1 yields

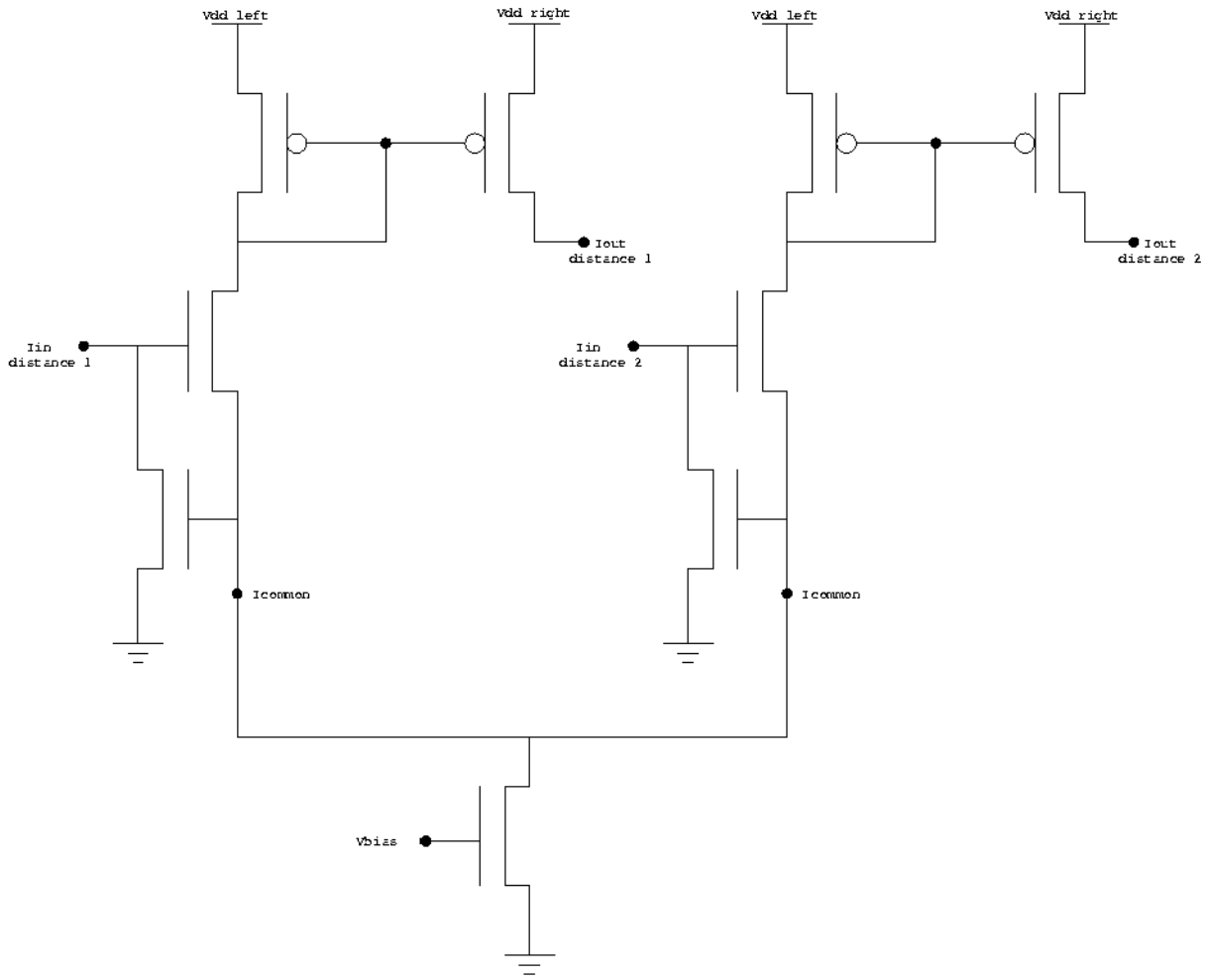
$$V_1 = V_o \ln(\frac{I_m + \delta_i}{I_o}) + V_o \ln(\frac{I_c}{I_o})$$

The winning output encodes the logarithm of the associated input. The symmetrical circuit topology ensures similar behavior for increases in I_2 relative to I_1. (Lazzaro et al. 1988: 2-3.)

I have done two small modifications to their basic design. Instead of having direct coupling to the power supply on the sources of transistors T_{2_1} and T_{2_2} (Figure 20) I add current mirrors, as can be seen in Figure 21. This modification is done to provide current outputs, instead of voltage, as the following neurons take a current as input. These current mirrors have separate power supplies for the left and right PMOS transistors. This is done for the same reason as in the current mirror, to provide a means of amplification adjustment. The following neuron encodes the current on each disparity line in the frequency of firing. The following AER sender multiplexes these spikes onto a bus. The bandwidth of the bus is limited, and so there may be collisions in time. The AER sender handles this by arbitration, i.e. if there is a collision one of the spikes gets delayed (until the first one is handled.) To avoid having too many collisions one may lower the amplification of the WTA current mirrors. By lowering the current outputs one also lowers the frequency of firing in the output neurons, thus lowering the number of collisions in the AER sender.

A simulation of the WTA with two inputs and outputs is shown in Figure 22. The final implementation has 65 inputs and outputs, as there are 65 disparity planes.

Figure 21 The winner-take-all circuit. The "Iin distance 1" and "Iin distance 2" inputs are the



currents from two different disparity planes. By adjusting the voltage on Vbias one alters the amount of current flowing through the common wire. Larger current will mean a faster decision of right disparity, but it may also lead to an unstable output because of increased sensitivity to noise. The bias has to be tweaked for optimal performance. (You can add as many WTA elements as you need by connecting all the Icommon nodes to the common NMOS transistor.)

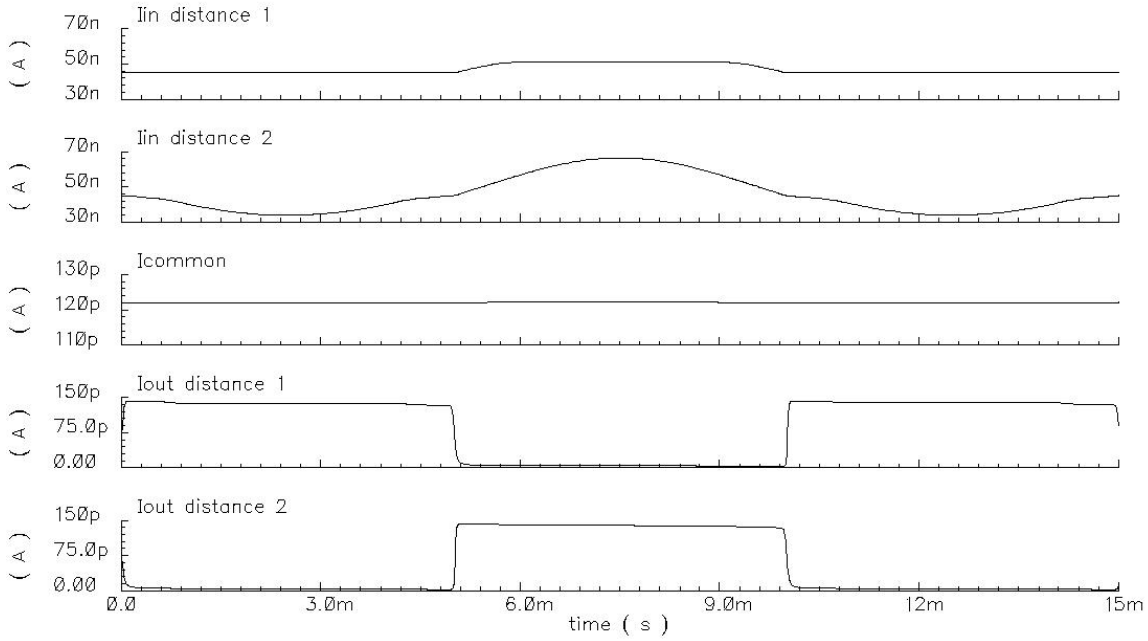


Figure 22 Simulation of the WTA circuit in Figure 21. (The final implementation has 65 inputs and outputs, but is otherwise equal to the one shown here.) Current source 1 is steady around 50nA (45 – 51nA) and current source 2 is sinus-shaped with a frequency of 100Hz (50nA ± 16nA). When the current of source 2 is below 1, output 1 goes high while output 2 is effectively cut off. The reverse happens when current source 2 goes above 1.

5.5 OUTPUT NEURON

As mentioned earlier the output neuron transforms the current output of the WTA to a stream of spikes suitable for encoding in the AER sender. There is one neuron per disparity plane. The neuron is self-resetting so it will encode the level of current in the frequency of firing. The neuron is a standard self-resetting integrate-and-fire neuron as shown in Figure 23.

The capacitance C1 integrates the input current until the switching threshold of the inverter I1 is reached. This makes the output voltage of I1 sink which in turn makes the output voltage of I2 rise. The output of I2, which is also the output of the whole neuron, is fed back through capacitor C2. This makes for a positive feedback, which pulls the voltage on C1 up (, by about $\frac{C2}{C1+C2}\times V_{dd} = \frac{1}{2}\times 5V = 2{,}5V$ in the final implementation.) The series connection of the two inverters is in reality a high gain amplifier which, when the output voltage Vout is fed back through C2, makes the transition from low to high happen very fast. (For an in-depth explanation see (Mead 1989: 198-201.))

The output voltage Vout is also fed back to transistor T1 that closes the flow of input current. It also opens transistor T4; and this makes the charge on C1 drain out through transistors T3 and T4. T3 is added to control the rate of discharge, thereby allowing control of the length of the output pulse/spike.

Transistor T2 is added for leakage purposes. The WTA will not suppress the wrong disparities totally, and so after a while even the disparity planes with very low current outputs from the WTA might charge capacitance C1 above the threshold of inverter I1. This is a sort of noise we would like to suppress. By having a slight leakage of current from capacitance C1 we suppress these "false" outputs.

A simulation of the circuit is shown in Figure 24.

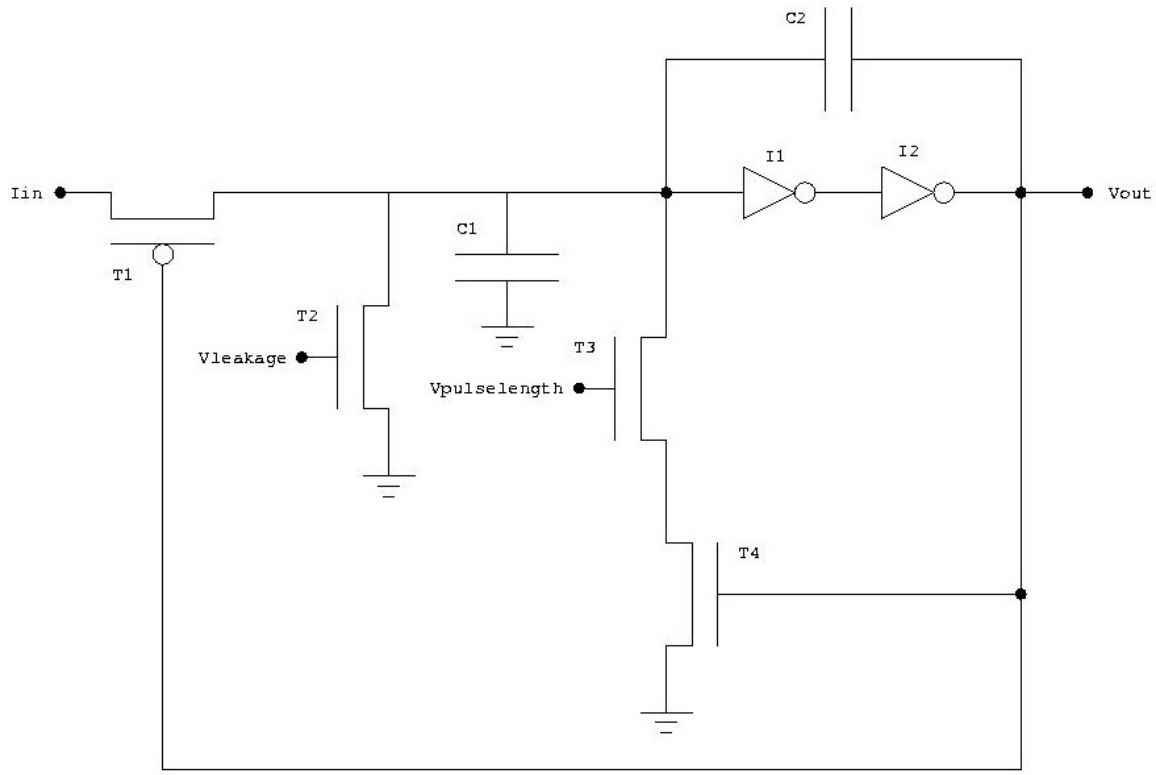


Figure 23. Self-resetting integrate-and-fire neuron. (Designed by Carver Mead of the "Computation and Neural Systems Group" at the California Institute of Technology (Mead 1989: 198-201.))

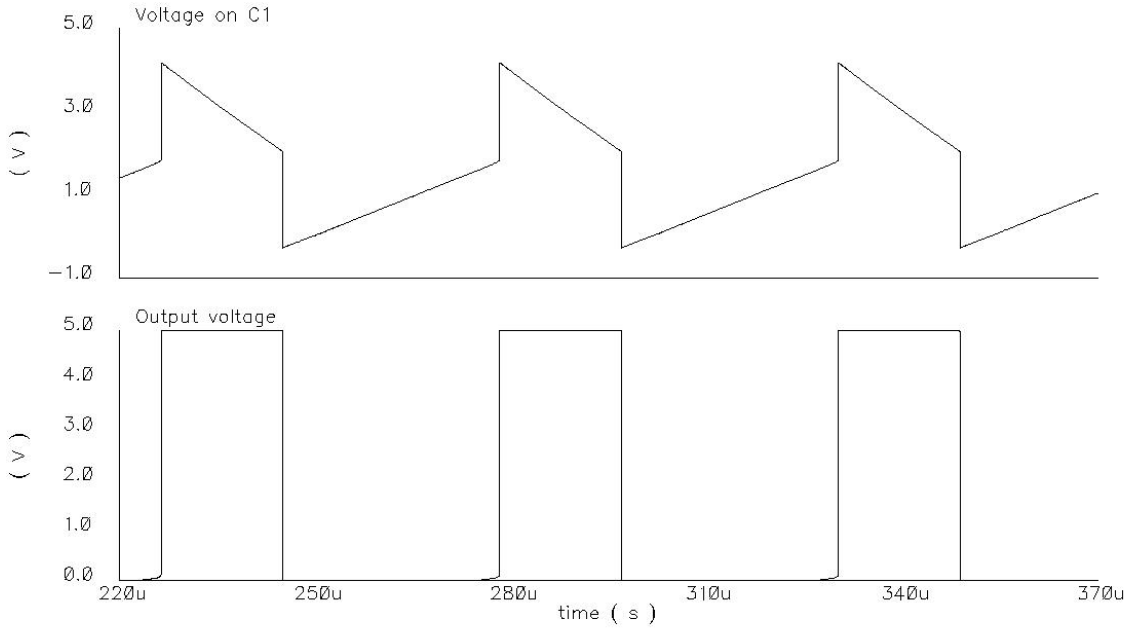


Figure 24 Simulation of the integrate-and-fire neuron of Figure 23. The neuron is fed a suitable, steady input current. As the simulation shows the inverters of the neuron, with the positive feedback added, will shape the varying voltage on C1 to a stream of pulses.

6 SIMULATIONS

We return again to the flowchart of Figure 14, but this time I have added all the appropriate biases. For understanding the total system dynamics I have also added the type of signal used from part to part, see Figure 25.

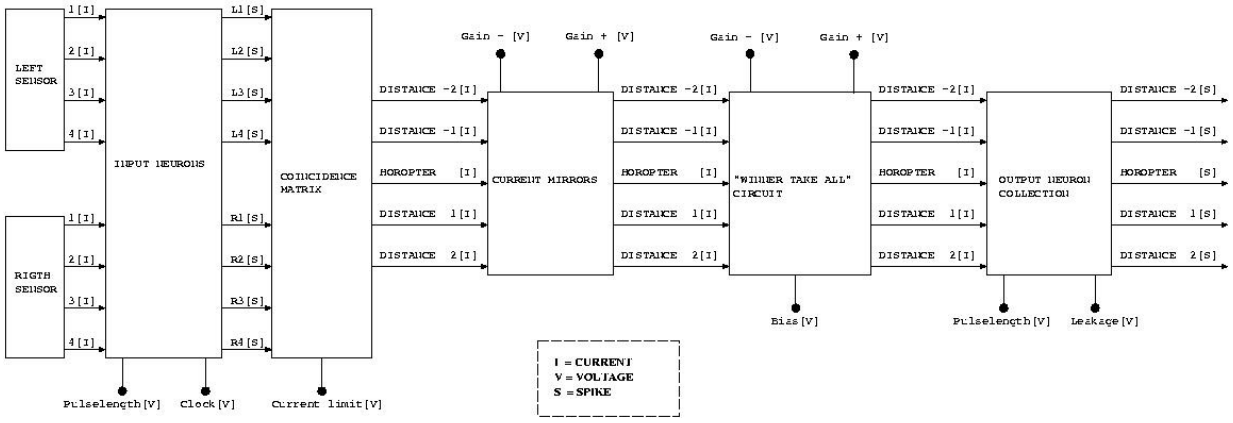


Figure 25 Simplified flowchart of the system. The real system has 64 pixels on each sensor and 65 disparity lines, but the principle of operation is the same. The figure include all the biases we can tweak to optimize system performance.

We tweak all these biases, as described for each individual part in chapter 5, to give the optimal system performance. A simulation of a smaller version of the final system, without the AER communication circuitry, is shown in Figure 26 and Figure 27. The system simulated has 16 pixels in each sensor and 9 visual disparity planes, but is otherwise equal to the final implementation. In the figures only the two disparity planes with the most coincidence events are shown. Disparity plane "–4" is the right answer.

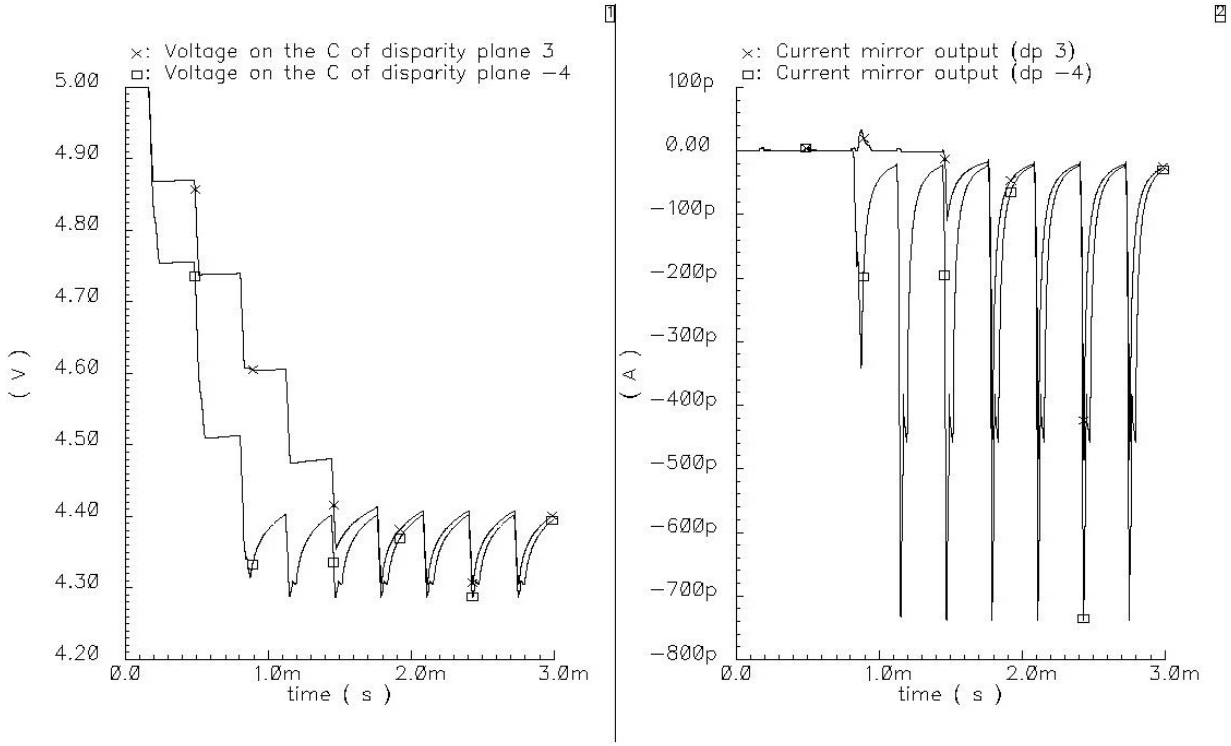


Figure 26 A simulation of a smaller version of the final system without the AER communication circuitry. (Clock period 0.32 ms). Only the two disparity planes with the highest number of coincidence events are shown. (The stimuli used were a set of PMOS transistors with random sizes of drain current.)

The left figure shows the voltage on the capacitance of the current mirror in the two planes. There are several coincidence events during each period and so the voltage of each of the capacitances is lowered. (A coincidence event causes a drain of current.) The amount of current is proportional to the number of coincidence events and the timing of the correlating spikes, as explained earlier. Since -4 is the right disparity, this plane has the highest number of coincidence events, and so the voltage on its capacitance is lowered quicker than the others. When the voltage reaches 4.4V, during the third period, the current output of the current mirror increases substantially as can be seen in the right figure. (The threshold of the PMOS transistor in the current mirror has been reached, i.e. the Vgs < -0.6V.) After the last coincidence event during a clock low, the voltage on the capacitance will increase exponentially due to the current flowing into the capacitance from the left PMOS of the current mirror, see Figure 19. (Due to the very limited size of a capacitance implemented in CMOS VLSI this will happen quite fast.)

The right figure shows the output of the current mirrors. The polarity of the current in this simulation is inverted compared to the current of the final implementation. (The size of the current though is the same.) During the fifth period the voltage on the capacitance of disparity plane 3 also reaches 4.4V, and so its corresponding current rises, but never above the current of disparity plane -4. Nonetheless, it is a task for the following WTA system to suppress all currents of false targets.

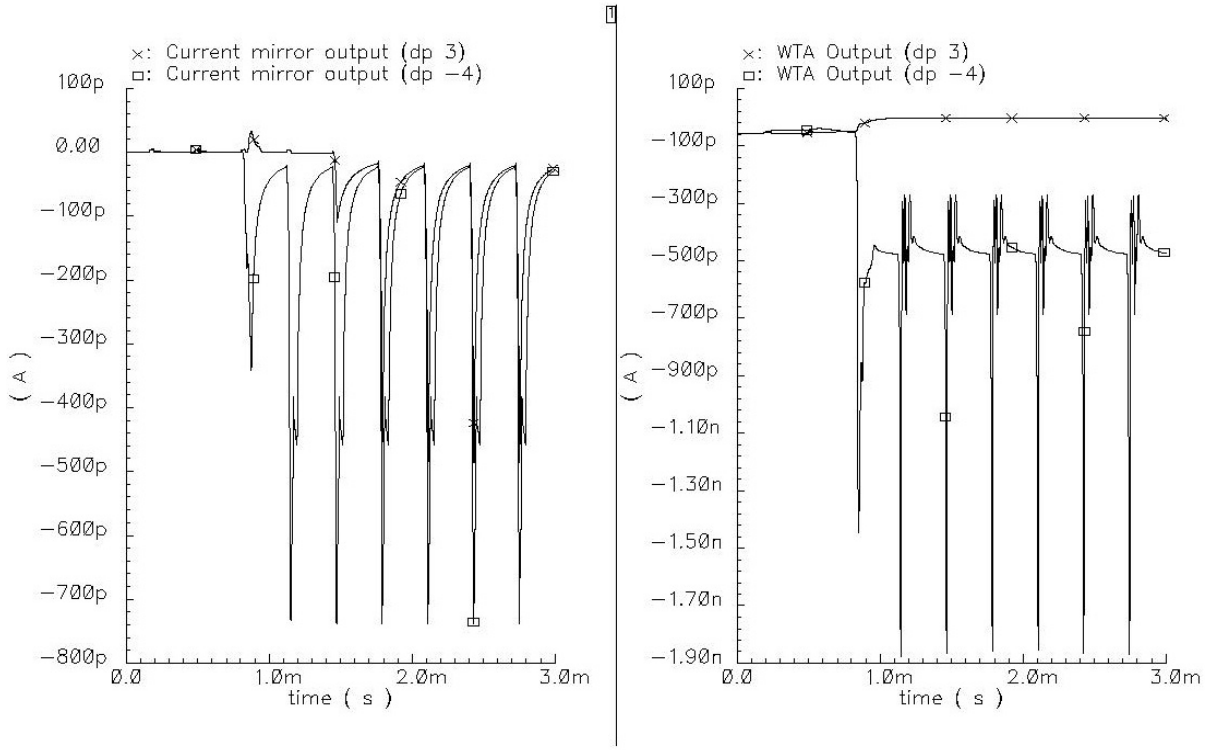


Figure 27 A simulation of a smaller version of the final system without the AER communication circuitry. (Clock period 0.32 ms). Only the two disparity planes with the highest number of coincidence events are shown.

The left figure shows the output of the current mirrors, the same as the right part of Figure 26. Both disparity planes have approximately the same current until the third period. Therefore the output of the WTA, shown in the right figure, is approximately the same on each line as explained in chapter 5.4. But, when the current of disparity plane -4 increases substantially during the third period the WTA suppresses all other lines and only relegates the current of this plane.

Most of the preliminary simulations before the actual VLSI layout design was carried out on a system with 16 pixels in each sensor and 9 visual disparity lines, as the one used in Figure 26 and Figure 27. This is of course limited in comparison to the final implementation, which uses 64 pixels in each sensor and 65 visual disparity planes. As I stated already in the introduction (, see chapter 1.5,) the simulation of analogue VLSI circuits are very computation-intensive. A 2ms transient simulation of the complete final implementation, without the AER communication circuitry, took over 21 hours to complete on a modern Unix workstation[7]. So, the only simulations carried out on a full-scale system were the final rounds to ensure the system worked in full-scale.

[7] Workstation: Sun Ultra 10 with 440MHz UltraSparc-II cpu. CPU utilization: 99,6%. Software: Cadence simulation with SpectreS ver.4.4.3.100.35.

7 VLSI CHIP LAYOUT

The layout was made for an 84-pin chip produced with AMS $0.6\mu m$ mixed signal process.

The layout of the chip was done with the following factors in mind:

- Process variation: A minimum size transistor will, according to percentage, vary more in size compared to a larger one due to process variations. This means the drain current, with some set gate voltage, will also vary more on small transistors compared to larger ones. It is important to have fairly good conformity between different system components of the same type in this analogue system. If for instance a particular current mirror has a much larger output current due to some set voltage on its capacitance (see chapter 5.3) than any other, it is likely that the corresponding disparity plane will have a predisposition to be "the winning" disparity of the WTA system. Even though it is possibly the wrong disparity. Minimum size transistors are therefore only used in digital parts like the RS-latch of the input neuron (, see chapter 5.1), and the NAND gates of the coincidence detectors (, see chapter 5.2.) As long as the transistor is used as an analogue component with continually variable drain current it is substantially longer than a minimum size transistor. I also avoid minimum size transistors wherever they are used as current limiters.

- Size limitation: Every single part of the circuit is custom designed to minimize its size. On the final chip we end up with some area free, but this only means we could implement a more precise system, i.e. having more disparity planes, on a similar chip using the same process. (A fairly uncomplicated scaling of the present design could do this.)

- Ease of routing: Due to the parallel nature of the system it will consist of a large amount of cross-connected similar components, especially in the coincidence matrix. It is therefore of utmost importance to stress the ease of routing when designing each of these components.

- Logical construction: The placement of the different components is structured and roughly follows the flowchart of Figure 25. The shortest route between two points is a straight line, and so the flowchart is a roadmap to the best routing. I have also wanted to make the layout as well arranged as possible for ease of later corrections and optimizations. (The strictly hierarchical and logical construction of the layout is the two key factors for making the system easy to scale up or down in size.)

Since we use temporal coding of the latency type the clocking of the system is important. All input neurons must be reset at the same time to have a common time reference. Optimal routing of the clock signal would involve H-tree geometry, but is not done in this layout. The length of a spike is in the order of ms and the distribution of the clock is in the order of ns. Therefore it is not necessary to use H-tree geometry. For the operation of this circuit,

with the low clock frequencies employed here, the difference in time-of-firing between two input neurons with a specific input current will be approximately the same.

Standard circuitry surrounds the presented system to provide input and output

I have added some test nodes to be able to check the performance of the chip more closely than can be seen by the AER output. The test nodes are scattered throughout the system to be able to follow the flow of signals from the AER input to the AER output along the two visual disparity planes 31 and 32. I have also added some individual test circuits to be able to assess the performance of each individual building block of the visual disparity computing system (, see Appendix B: 11.2.) . A plot of the layout is shown in Figure 28.

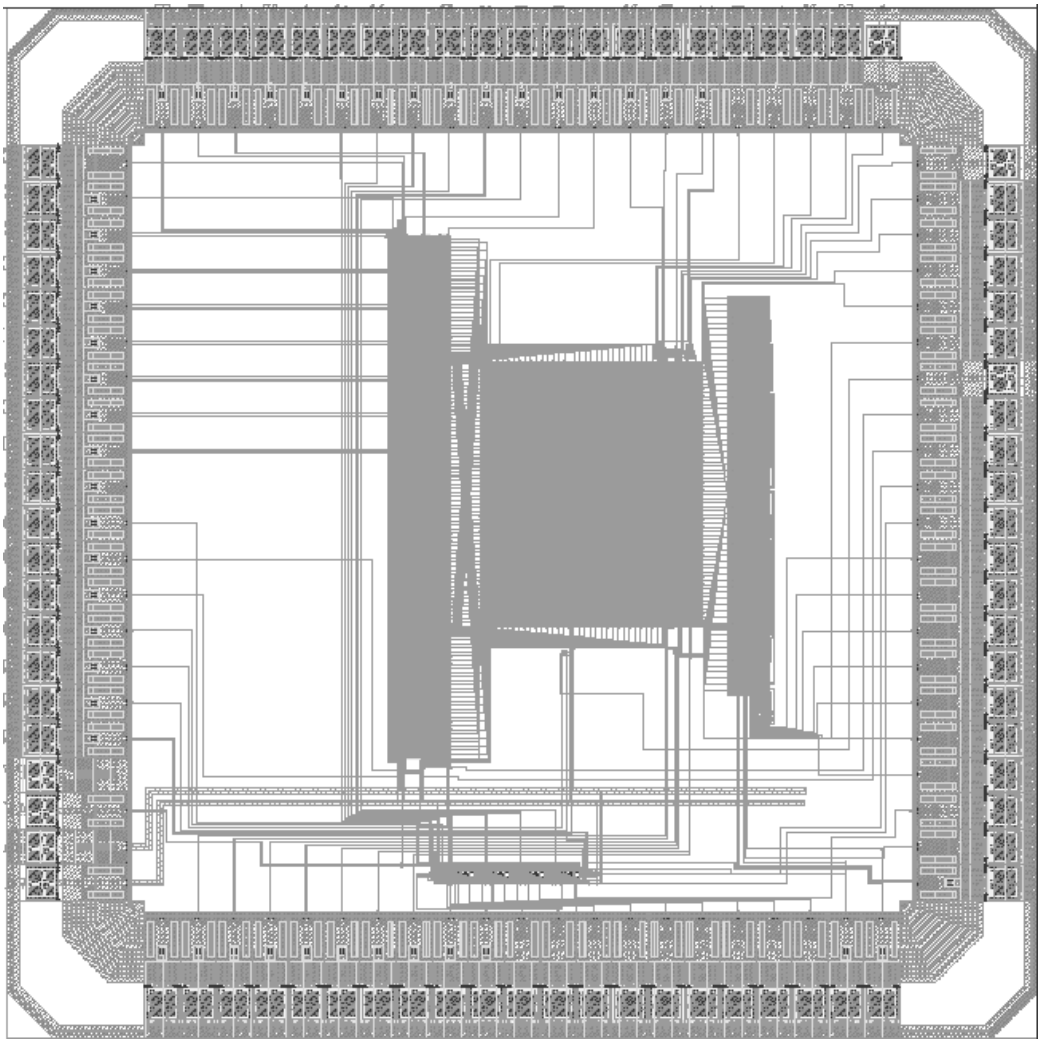


Figure 28 The VLSI implementation in the AMS $0.6\mu m$ *mixed signal process.*

8 MEASUREMENTS AND DISCUSSION

In the final test runs the chip was tested with input that resembles both static and dynamic images. The static input was a ramp stimulus with a positive gradient as illustrated in Figure 29, or a negative gradient as the one used for the experiment of Figure 31. The figure shows the conceptual images from the left and right sensor, while the actual input to the chip is an AER conversion of the average frequency of the individual pixels with Poisson distributed spikes. The following mapping of pixel nr to AER address is used for the onboard AER receiver:

- L1 = 127, L2 = 125 … L63 = 3, L64 = 1
- R1 = 0, R2 = 2 … R63 = 124, R64 = 126

The AER onboard sender uses the following mapping of disparity plane (dp) to AER address (a):

- dp -32 = a 64, dp –31 = a 63 …. dp -1 = a 33, Horopter = a 32, dp 1 = a 31 …. dp 31 = a 1, dp 32 = a 0

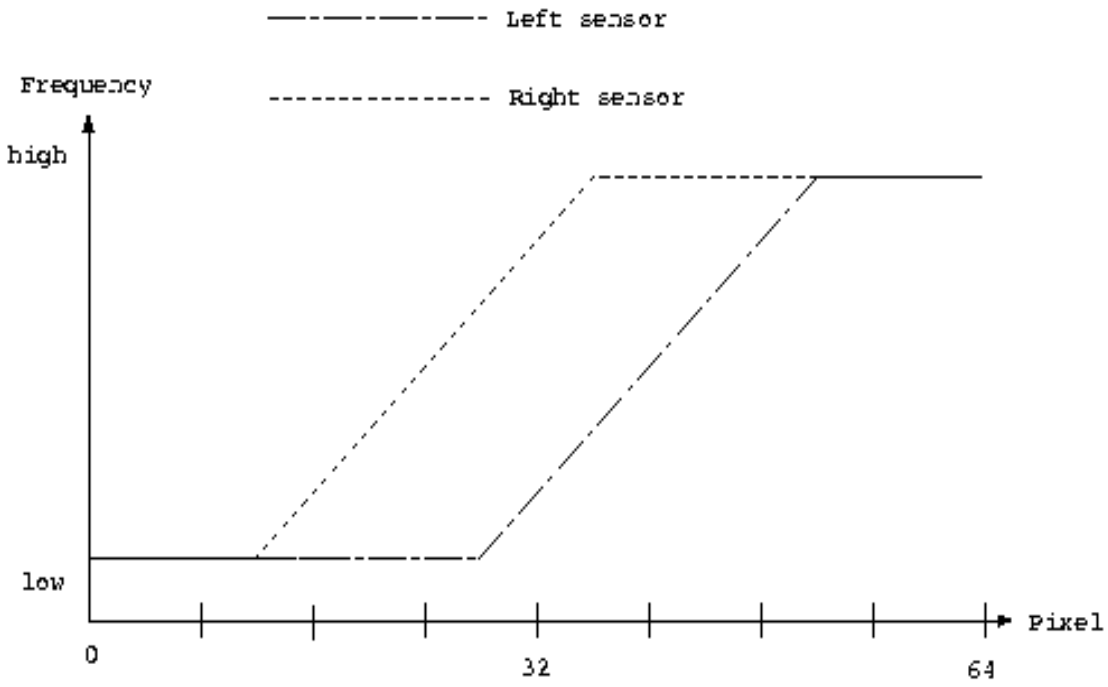


Figure 29 Ramp stimuli with a positive gradient. The right sensor image is a shifted version of the left sensor image. The center of the right image is shifted 8 pixels to the left, and the left image is shifted 8 pixels to the right. This makes disparity plane 16 the right output from the chip. The image in question would be of a level surface with a linearly increasing intensity from left to right with a stretch of constant intensity at the left and right side. The leftmost pixel of the image would be quite dark while the rightmost pixel would be bright. (The average

frequency of firing constitutes the coding of intensity level. This rate coding is converted to temporal coding by the one-shot integrate-and-fire neurons of the chip.) The addresses put on the AER bus is Poisson distributed. (The frequencies on the y-axis are purely an example.)

Figure 30 is a histogram of the AER output due to a ramp stimulus input. The ramp is identical to the one in Figure 29 except for the fact that there is no shift. AER address 32 is therefore the right winner, since it corresponds to the horopter. All biases have been tweaked for optimal performance. The chip chooses AER address 30 as the right answer.

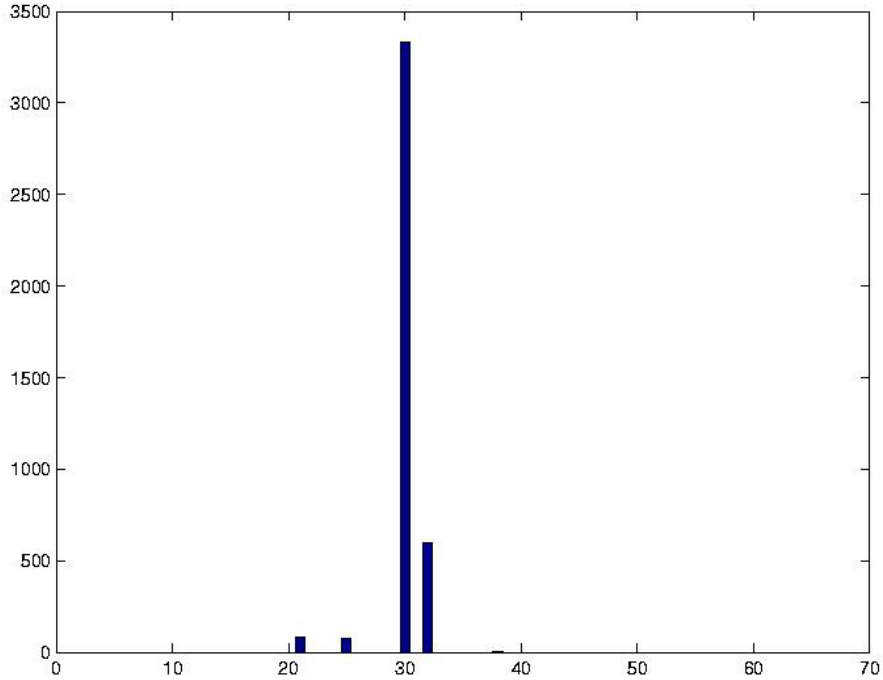


Figure 30 Histogram of the chips AER output due to a ramp stimulus with positive gradient. Y-axis shows number of AER events and X-axis shows AER address. The right answer is 32. The chip comes pretty close to this by choosing 30 as the right winner with 32 as the runner-up.

For most static input images, including step and ramp stimuli with different shifts, it is possible to tune the chip to give the right answer, or one very close to it. So the main principles of operation also work in a physical VLSI implementation.

Next I tested the chip with the same set of biases but with a different input. The input was now a ramp stimulus with a negative gradient and no shift. AER address 32 should therefore still be the winner. As Figure 31 shows the chip outputs the wrong answer. Address 21 is the winner with 49 as the runner-up. Address 32 is only the fourth most frequent address on the AER output bus. This problem was also evident in other tests with static input; that no set of biases was optimal for different inputs.

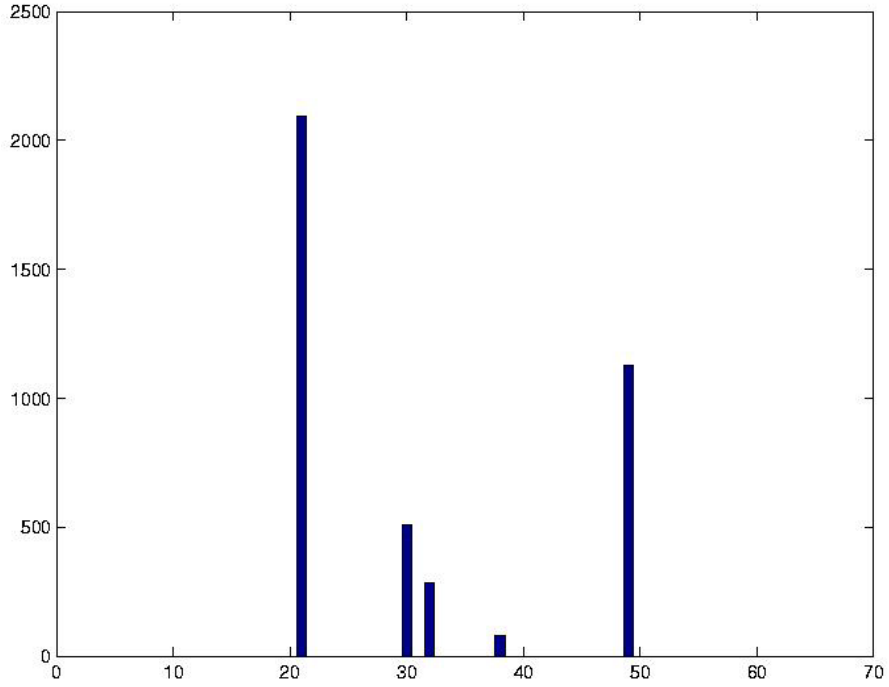


Figure 31 Histogram of the chips AER output due to a ramp stimulus with negative gradient. Y-axis shows number of AER events and X-axis shows AER address. The right answer is 32.

To give a good assessment of the performance of the chip with some specific set of biases I tested the chip with dynamic input. The input was given as a step function, as shown in Figure 32. The step was itself modulated by a stair function of 25 different shifts in 5 seconds. The output is shown in Figure 33.

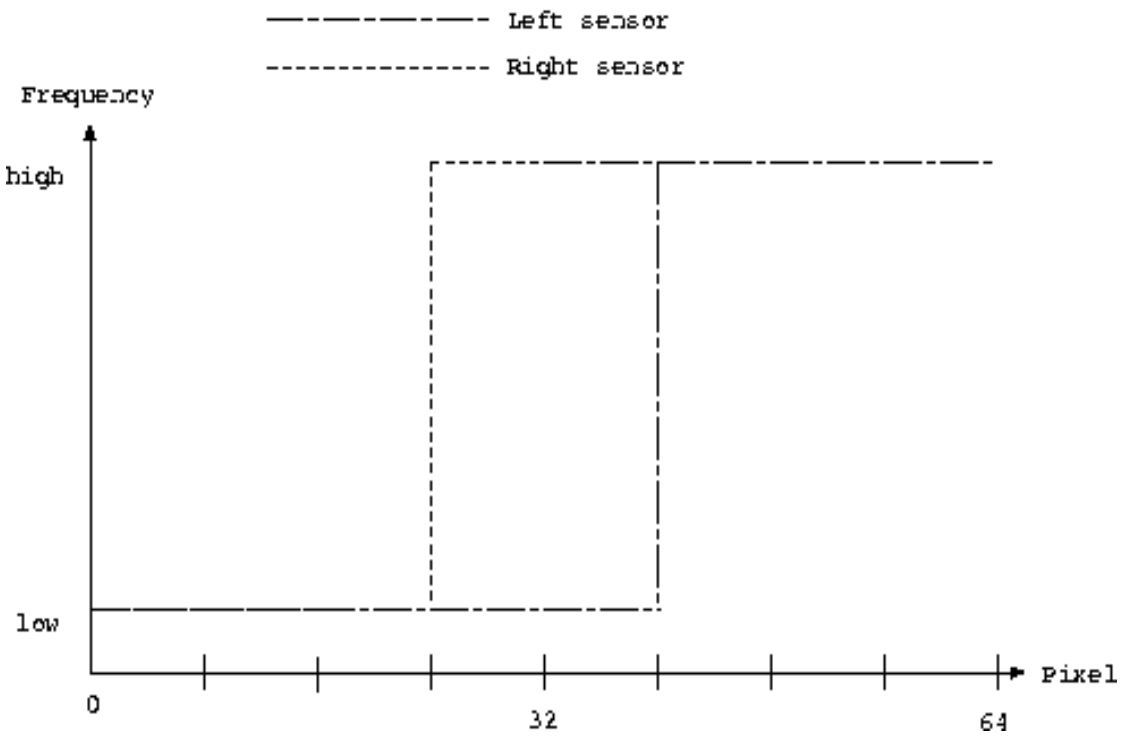


Figure 32 Step stimulus with a positive gradient. The center of the right image is shifted 8 pixels to the left, and the left image is shifted 8 pixels to the right. This makes disparity plane 16 the right output from the chip. The image in question would be of a level surface with constant stretch of low intensity on the left and a constant high intensity on the right. The pixels to the left of the step would be quite dark while the pixels to the right would be bright.

The dynamic test in Figure 33 shows the same tendencies as the static test of Figure 31, that address 21 is very prominent. In Figure 34 only the AER addresses put on the bus while the clock is low are shown, i.e. when the chip is not reset. Address 21 is not so prominent here, and this is because it is a "natural winner". Because of process variation the matching of the individual elements of the system is not very good. Therefore some of the disparity planes become natural winners, i.e. disparity planes that seem to have larger currents flowing into the WTA than other planes even though the number of coincidence events in the matrix is lower on this plane. This is made more evident during chip reset. The size of capacitances on the VLSI chip is very limited and the integration of current in the current mirrors is therefore not terribly good. Since the WTA also works fast it has a tendency to return to some natural winners/attractors when the chip is reset and there is very little current input from the other disparity planes.

The rest of the AER output events seem to be the same in both Figure 33 and Figure 34. Among these are also some lesser attractors, most notably address 40. This attractor is probably due to a mismatch in some other part of the circuit than the case is for address 21.

The reason might for instance be that the current from each coincidence event in one of the coincidence detectors on disparity plane –8, is bigger than from other coincidence detectors. (Disparity plane –8 corresponds to the AER output address 40.) A transistor mismatch could cause such a problem. A mismatch in this part of the circuit would only be evident when the coincidence detector in question outputs current. This will only happen when the right answer should be close to address 40 and the clock is low. This is the case for the lesser attractor with address 40.

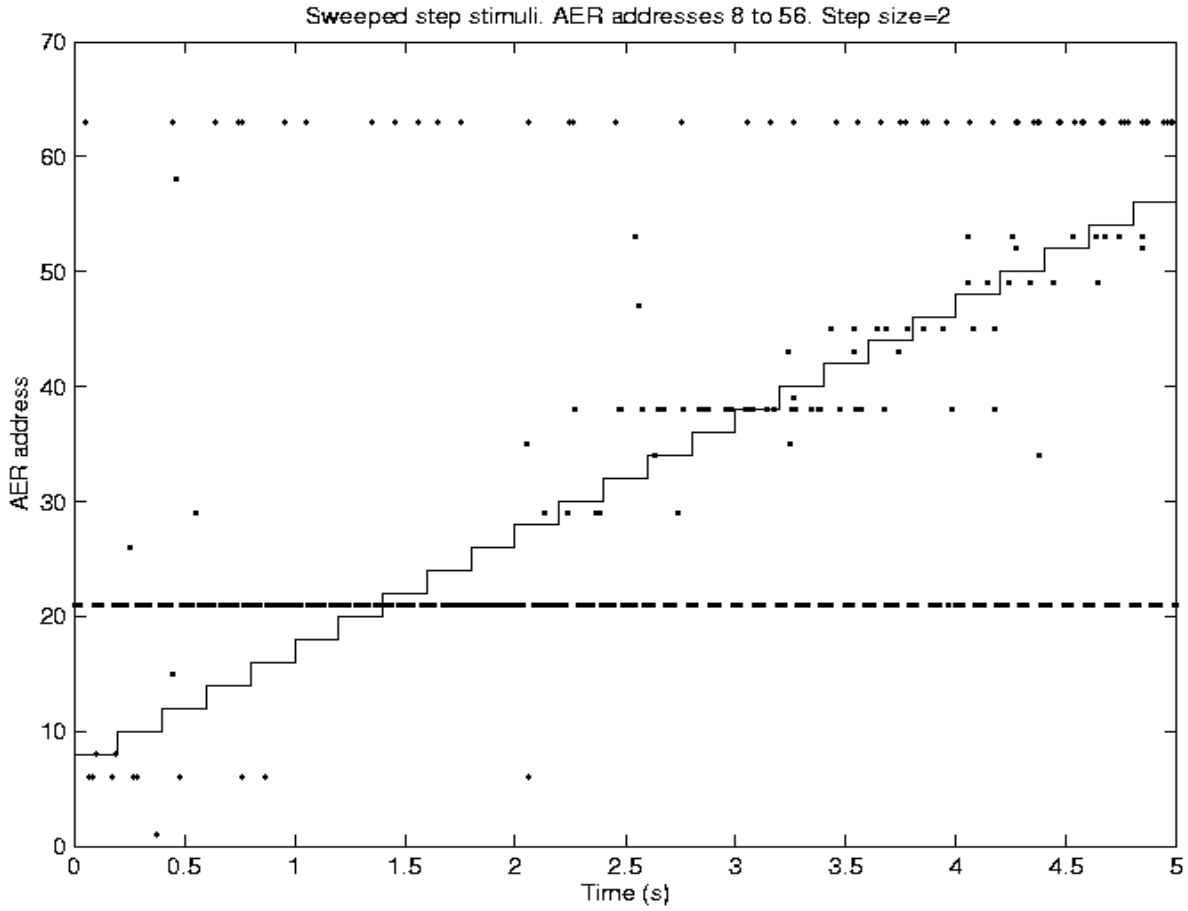


Figure 33 Chip performance with swept step stimuli with a positive gradient as input. The stair function shows the correct answer to the input at any time while each dot marks an actual AER event with a specific address at some specific time. Address 21 is "attractor" or what we might call natural winners. Some local attractors are also evident, most notably address 40.

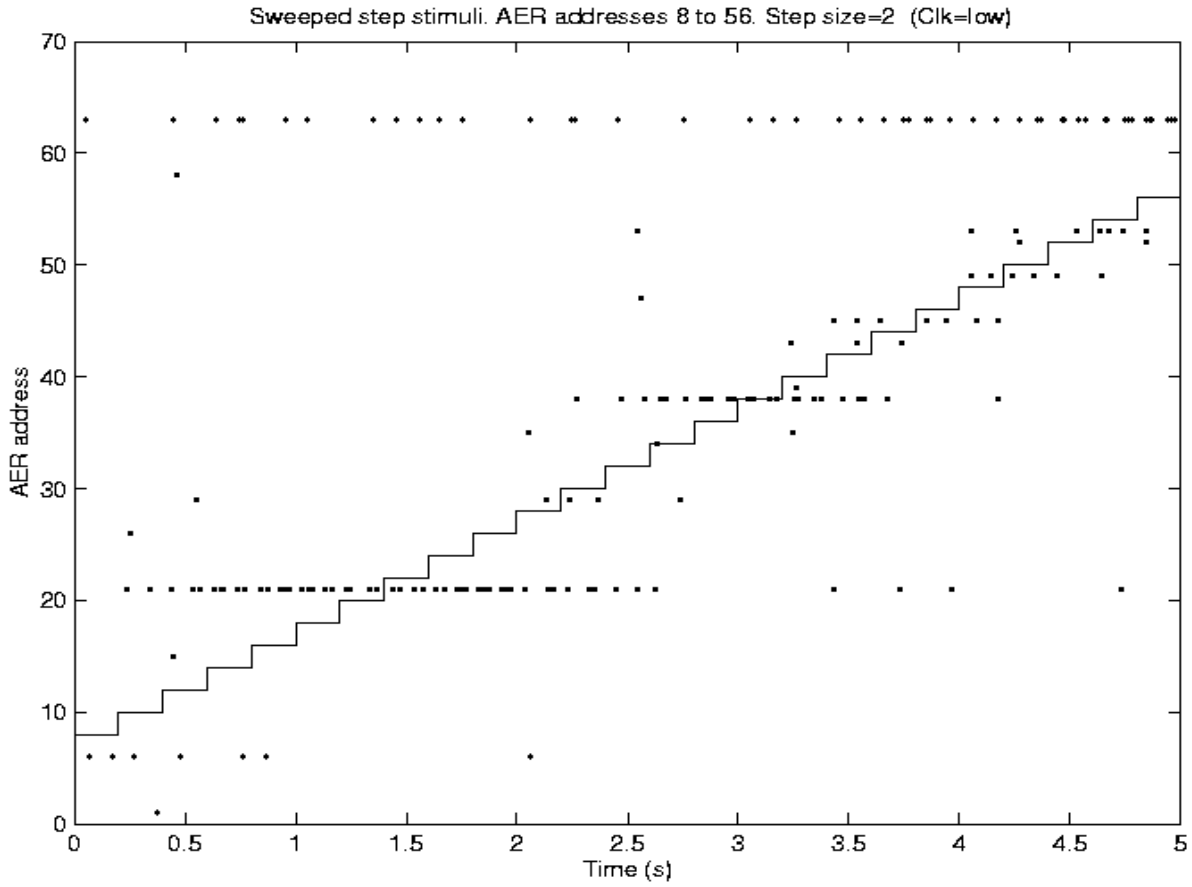


Figure 34 Chip performance with swept step stimuli with a positive gradient as input. The stair function shows the correct answer to the input at any time while each dot marks an actual AER event with a specific address at some specific time. The figure only shows addresses put on the bus while the clock is low, i.e. while the chip is not reset. Address 21 is not quite so prominent in this figure as in Figure 33.

I tested several different identical chips with the same input to eliminate the possibility of the natural winners being a result of inferior layout. None of the chips had the same addresses as attractors, and the strength of these attractors varied from chip to chip. Figure 35 shows exactly the same experiment as the one in Figure 33, but with another chip. Address 5 is the strongest attractor here, but clearly this chip shows a better performance. In Figure 36 I have again removed the events put on the bus when the clock is high. The frequency of firing of address 5 is now very much damped. This chip shows the best performance of the five tested.

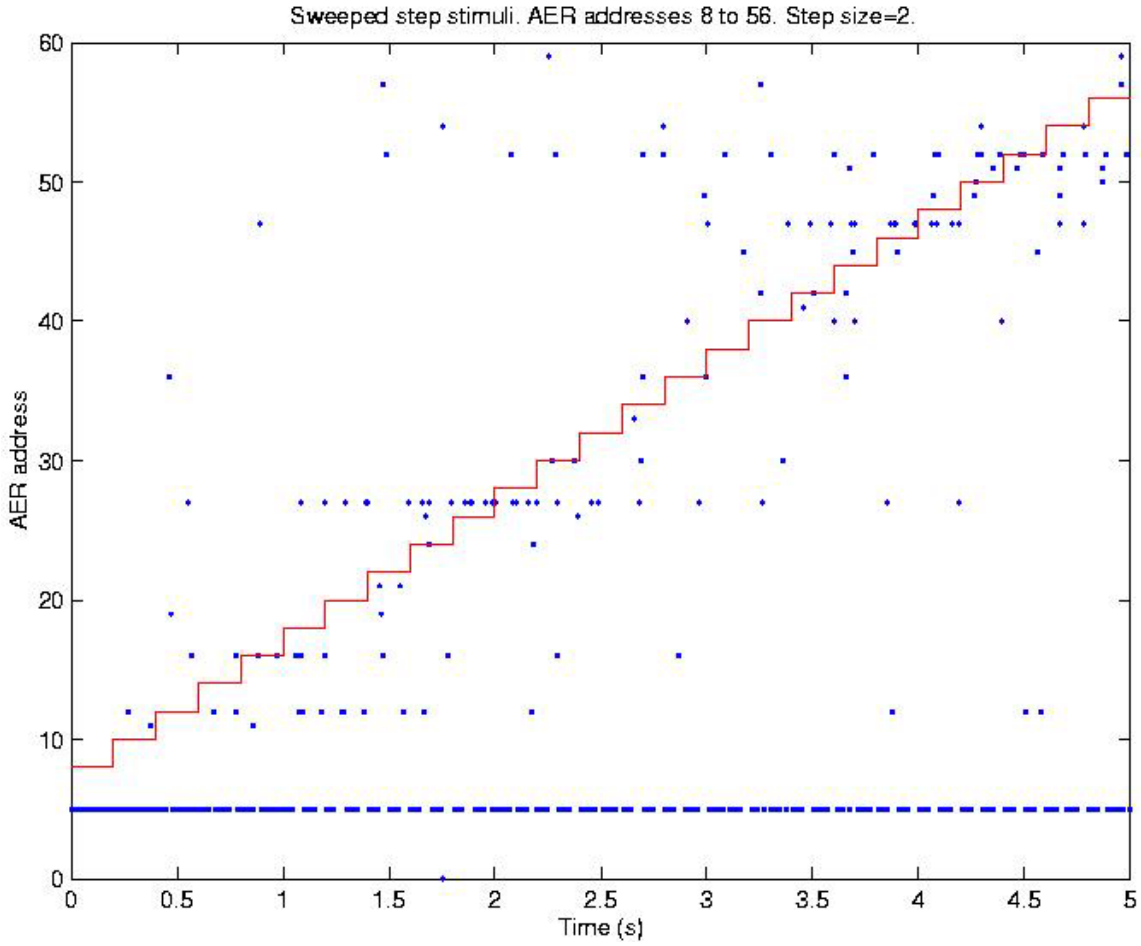


Figure 35 Chip performance with swept step stimuli with a positive gradient as input. (Chip nr.2.) The stair function shows the correct answer to the input at any time while each dot marks an actual AER event with a specific address at some specific time. Address 5 is an attractor.

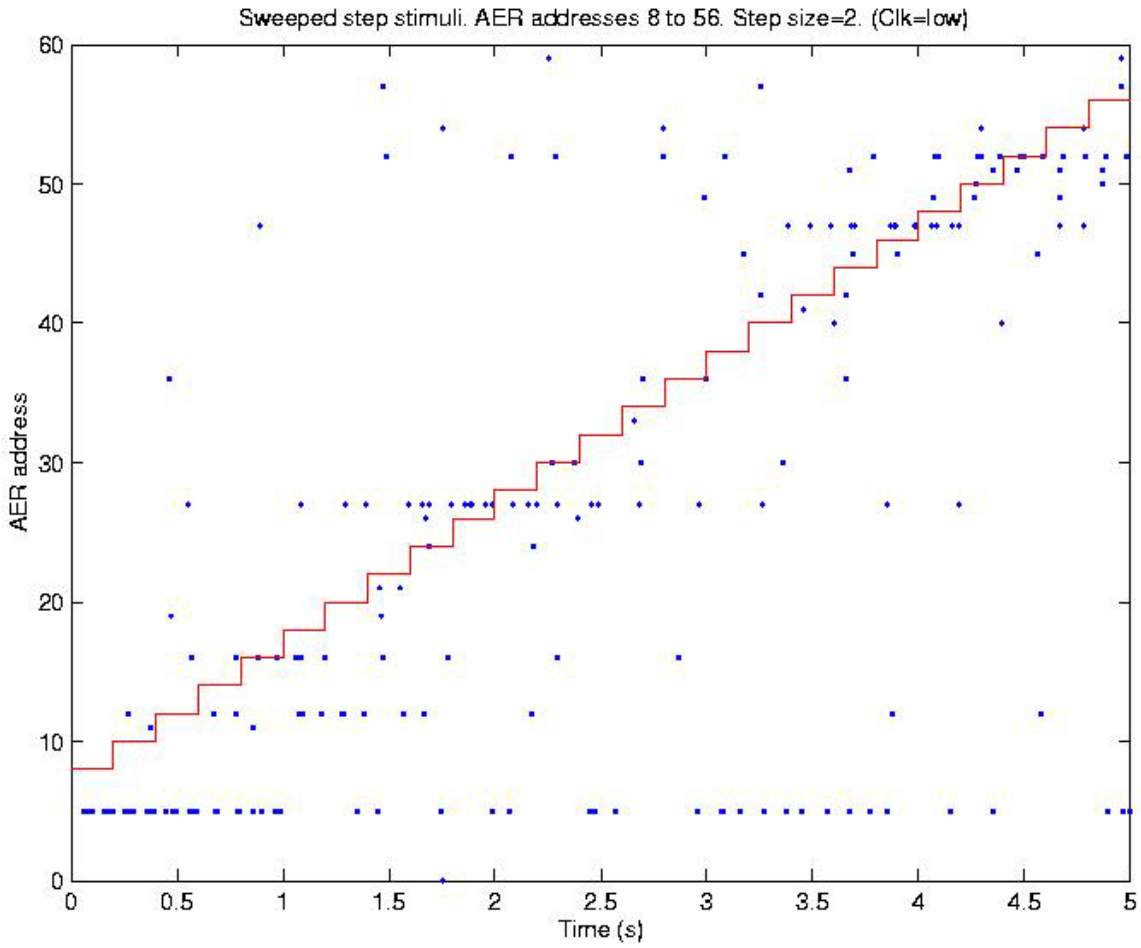


Figure 36 Chip performance with swept step stimuli with a positive gradient as input. (Chip nr.2.) The stair function shows the correct answer to the input at any time while each dot marks an actual AER event with a specific address at some specific time. The figure only shows addresses put on the bus while the clock is low, i.e. while the chip is not reset. Address 5 is an attractor, but it is clear that it is most often the winner when the clock is high. (Ref: Figure 35.)

Taking a look at the measurement seems to make two things quite clear:

- Temporal coding *is* an efficient way to code the intensity of the pixels of an image and *does* provide an easy way to compare intensity levels.

- The organizing principles of chapter 1.3 should be investigated further.

It is in the implementation of the organizing principles that this chip has the biggest potential of performance improvement, especially in the forms of adaptiveness and

redundancy.

The problem of mismatch of individual building blocks, like transistors, due to process variations is the most important factor that impedes the performance of this chip. By having more redundancy, for instance by using several current mirrors in parallel instead of one, we will get better matching of current between the different disparity planes. One could also do this by just making larger structures, as they will give better matching. I had this fact in mind when doing the layout, as mentioned in chapter 7, but even *larger* structures should be implemented. In the specification papers for the AMS process in use there is measurements of the process variation data. The data show that larger transistors *will* have better matching, even between two identical structures located close to each other in the layout (AMS 1998: 29).

During testing of this chip I had to tweak the biases quite a lot to get the best performance. When having dynamic input we have to rely on some specific set of biases, and as the measurements shows this is not optimal. So, adaptiveness in the form of self-adjustment of the biases is needed. This is even more the case when using actual cameras for real-world image input where we have little control of the input (Mead 1989: 72). This is an area of extensive research because it is a general problem for many similar analogue circuits within the area of neuromorphic engineering.

9 CONCLUSION

A visual disparity computing system implemented in analogue VLSI has been proposed, developed, simulated, implemented and tested. In theory it can it can compute the right visual disparity given two input images/stimuli. The actual aVLSI implementation has proven to have some limitations, but the chip can still compute the right visual disparity for most static images; after some initial fine-tuning of the biases.

Temporal coding has proven to be an efficient way for encoding the intensity level of a pixel in a visual sensor. It also provides an easy means for comparing the intensity levels of two pixels. Our goal of an effective "representation of information" (, see chapter 1.4,) has been reached.

As I stated in the chapter on materials and methods (chapter 3) there are a number of problems inherent in the real world of VLSI chips, not least process variations, which is hard to simulate on computer software. So the ultimate test/judge would have to be real experiments on a physical chip. The attractors, being disparity planes prone to win even though they are not the right answer, is a result of this process variation. It did not manifest itself in the simulations of chapter 6. Thus the circuits of neuromorphic engineering needs to be implemented in analogue VLSI and exposed to physical test runs to show their true strengths and weaknesses.

To heighten the level of performance and robustness of the chip, one has to look into ways to get better matching of each individual building block, for instance using larger structures, and mechanisms for adaptation to different input stimuli. Both these improvements sort under the "organizing principles" of the nervous system (, see chapter 1.3.) The implementation of this chip has made it even clearer, as most research in the area of neuromorphic engineering has, that finding out which, why and how these organizing principles should be implemented is one of the biggest challenges of neuromorphic engineering. Since the ultimate goal of neuromorphic engineering is to produce chips that not only rivals, but also surpasses the performance of modern digital circuits, a better understanding of the organizing principles of the nervous system is essential.

10 BIBLIOGRAPHY

(AMS 1998) Austria Mikro Systeme International AG
"0,6μm CMOS CUP Process Parameters".
Document #: 99 33 011. Revision #: B. October 1998.

(Cariani 2001) Cariani, P.A.
Neural timing nets
Neural Networks nr.14 (2001) p737-753.

(Delbrück and Mahowald 1989) Delbrück, T., Mahowald, M.A.
Cooperative Stereo Matching Using Static and Dynamic Image Features
In Mead, C. and Ismail, M. (eds.), Analog VLSI Implementation of Neural Systems. Kluwer Academic Publishers, Boston. p. 213 - 238.

(Delbrück 2000) Delbrück, Tobi
Address-Event-Representation Resources
<http://www.pcmp.caltech.edu/aer/> Oct 24 2000

(Hoey 1998) Hoey, Jesse
The Perception of Depth
<http://webvision.med.utah.edu/KallDepth.html>

(Häfliger 2000) Häfliger, Philipp
Temporal Codes in Neuromorphic aVLSI
<http://www.ifi.uio.no/~vlsi/Hovedoppgaver/hafliger_hoppg00.html>

(Lane and Thacker 1996) Lane, R.A., Thacker, N.A.
Stereo Vision Research: An Algorithm Survey

(Lazarro et al. 1988) Lazzaro, J., Mahowald, M. A., Mead, C.A., Ryckebusch, S.
Winner-take-all networks of O(n) complexity.
In Tourestzky, D. (ed), Advances in Neural Information Processing Systems 1. San Mateo, CA: Morgan Kaufmann Publishers, p. 703-711.

(Lozman et al. 1997) Lozman, E., Rahemtulla, S., Schroeder, J.
Computer Vision - A 1997 Sophomore College Presentation
<http://cse.stanford.edu/classes/sophomore-college/projects-97/computer-vision/stereopsis.html>

(Marr and Poggio 1976) Marr, D., Poggio, T.
Cooperative Computation of Stereo Disparity
Science vol.194 (1976) p.283 - 287

(Mead 1989) Mead, Carver
Analog VLSI and Neural Systems
Addison-Wesley Publishing Company, ISBN 0-201-05992-4, 1989

(Mead 1990) Mead, Carver
Neuromorphic electronic systems
Proceedings of the IEEE, vol.78, no.10, oct.1990, p.1629 - 1636

(Thorpe et al. 2000) Thorpe, S.J., Delorme, A., Van Rullen, R., Paquier, W.
Reverse engineering of the visual system using networks of spiking neurons
IEEE International symposium on circuits and systems (ISCAS), p.IV-405 – IV-408

11 APPENDIXES

11.1 APPENDIX A: AER

During testing of the VLSI chip one needs to have a standard of communication for input and output. The visual stimuli used are synthetically produced for ease of testing and controlled laboratory conditions. This is done by using the software package Matlab. The output is analyzed by using the same software package. The input should consist of 2*64=128 channels/axons with a variable firing frequency for providing the "images". The output should consist of 65 such channels/axons. This large number of signals means we have to use some form of multiplexing of the signals/channels, since the VLSI chip only has 84 pins. These 84 pins also have to be used for biases (10), test nodes (21), clock (1) and power supply (9). There is also another independent circuit produced on the same VLSI chip that use some pins (22). (Philipp Häfliger of the Microelectronic Systems Group, Dept. of Informatics at the University of Oslo, has designed this circuit. The two circuits have been produced on the same die to cut production costs.)

I settled on the address event representation (AER) communications standard since it has been specifically developed for off-chip communication of spikes. Tobi Delbrück provides this description of AER:

> *The AER is a neuromorphic representation of signals. The idea is very simple: use the large bandwidth of metallic wires and silicon circuits to create a* virtual *bundle of biological axons, i.e., multiplex a large number of spike events onto a digital bus. A sender chip -- a silicon retina, for example -- has a bunch of neurons that can independently and asynchronously fire action potentials. When one of these cells want to send a spike event to some receiver neurons, it requests the bus. When it gets access, its address is placed on the bus. At the receiving end, the cells listen for when their address comes on the bus. When their address comes along, they give themselves a little pulse of charge simulating a postsynaptic potential. The important features of this communication scheme are:*
>
> *Time represents itself, meaning that there is no program counter or master clock: when a spike happens, its placement on the bus signals its occurrence.*
>
> *The bus capacity is efficiently utilized only to transmit "interesting" information.*
>
> *Spike latency is minimized. This feature goes along with the idea that precise spike timing is important.*
>
> *No spikes are lost to collisions. If multiple spikes happen simultaneously, they are transmitted as rapidly as possible in sequence. (Delbrück 2000)*

We are left with 21 free pins on the VLSI chip for the AER input and output circuitry. This is enough for a 7-bit address representation, enough for 128 channels of input and

output, plus the necessary signaling. (The signaling is a standard "4-phase handshake" receive and acknowledge system for both the sender and receiver AER circuits, a total of 4 bits.) 3 pins are left unconnected in the final implementation).

The AER receiver integrates each channel and has a current as output for each channel. This is due to the fact that we assume that each pixel of an actual sensor would have a current output that varies according to the intensity of light.

11.2 APPENDIX B: PIN LISTS

For reference purposes I have included these two pin lists. The first one, Table 2, shows all pins relevant for testing the complete system described in this paper. Table 3 shows pins connected to separate test circuits.

The test circuits have been included on the chip to be able to assess the performance of each individual building block of the visual disparity computing system. If the system as a whole did not work, these test circuits were meant to be used to pinpoint the source of the problem. This has not been necessary since the system as a whole roughly performs as simulated. Our largest problem (, see chapter 8,) of mismatches due to process variations cannot be studied using these test circuits. The test circuits are:

- One input neuron. PMOS used as current source. Pins: 3, 19, 46, 61 and 77.
- Two current mirrors and a WTA connected for assessment of WTA performance. NMOS used as current sources for the two current mirrors. Pins: 4,5,11 and 12.
- One output neuron. PMOS used as a current source. Pins: 17 and 48.
- One coincidence detector. Pins: 18, 59 and 60.

All test circuits use the same supply voltages and biases as the similar circuits in the main system except for the input neuron, which has its own bias for controlling the length of the spikes issued (Pin 46.) Each test circuit is identical to its match in the main system.

PIN NAME IN LAYOUT	NR	PIN NAME IN SCHEMATIC	DESCRIPTION
ANALOG<0>	2	rec_gain+	Supply for right PMOS of the current mirror of each pixel in the AER receiver.
ANALOG<1>	3	Vpulselength	Bias controlling the duration of the pulses/action potentials from the input neurons.
CONTROL<1>	6	Vbias	Bias controlling max total current flowing in the common wire of the WTA section.
DATA<0>	7	L33	Test pin. Voltage on the first input of a coincidence detector on disparity plane 32.
DATA<1>	8	R1	Test pin. Voltage on the second input of a coincidence detector on disparity plane 32.
DATA<2>	9	Vcap_mirror_d31	Voltage on the capacitance in the current mirror of disparity plane 31.
DATA<3>	10	Vcap_mirror_d32	Voltage on the capacitance in the current mirror of disparity plane 32.
DATA<6>	13	Vcap_n_d31	Voltage on the input capacitance of the output neuron of disparity plane 31.
DATA<7>	14	Vcap_n_d32	Voltage on the input capacitance of the output neuron of disparity plane 32.
DATA<8>	15	Vout_d31	Voltage on the input of the AER output circuit of disparity plane 31.
DATA<9>	16	Vout_d32	Voltage on the input of the AER output circuit of disparity plane 32.
AER_OUT<0>	23	snd_bit<0>	AER sender: Bit 0

AER_OUT<1>	24	snd_bit<1>	AER sender: Bit 1
AER_OUT<2>	25	snd_bit<2>	AER sender: Bit 2
AER_OUT<3>	26	snd_bit<3>	AER sender: Bit 3
AER_OUT<4>	27	snd_bit<4>	AER sender: Bit 4
AER_OUT<5>	28	snd_bit<5>	AER sender: Bit 5
AER_OUT<6>	29	snd_bit<6>	AER sender: Bit 6
AER_OUT_REQ	31	snd_req	AER sender: Request
AER_OUT_ACK	32	snd_ack	AER sender: Acknowledge
AER_OUT_PU	33	snd_pu_bias	AER sender: Pull-up bias
AER_OUT_PD	34	snd_pd_bias	AER sender: Pull-down bias
ANALOG<4>	47	Vleakage	Bias controlling leakage from the individual output neurons.
ANALOG<6>	49	WTA_vdd_right	Voltage supply for right pmos of all current mirrors in the WTA circuit.
ANALOG<7>	50	WTA_vdd_left	Voltage supply for left pmos of all current mirrors in the WTA circuit.
ANALOG<8>	51	CM_vdd_right	Voltage supply for right pmos of all current mirrors.
ANALOG<9>	52	CM_vdd_left	Voltage supply for left pmos of all current mirrors.
ANALOG<10>	53	foll-bias	Bias for a number of op-amp line drivers.
vdd!	54	vdd!	Global voltage supply.
gnd!	55	gnd!	Global ground connection.
PVDD	57	PVDD	Power supply for the pads of the chip.
ANALOG<11>	58	V_current_limit	Bias controlling the max current from each individual coincidence detector in the coincidence matrix.
CONTROL<4>	62	rec_timeconst	Current limiter for the integrator in the AER input circuit.
AER_IN<6>	66	rec_bit<6>	AER receiver: Bit 6
AER_IN<5>	67	rec_bit<5>	AER receiver: Bit 5
AER_IN<4>	68	rec_bit<4>	AER receiver: Bit 4
AER_IN<3>	69	rec_bit<3>	AER receiver: Bit 3
AER_IN<2>	70	rec_bit<2>	AER receiver: Bit 2
AER_IN<1>	71	rec_bit<1>	AER receiver: Bit 1
AER_IN<0>	72	rec_bit<0>	AER receiver: Bit 0
AER_IN_REQ	73	rec_req	AER receiver: Request
AER_IN_ACK	74	rec_ack	AER receiver: Acknowledge
AER_IN_PU	75	rec_pulse_length	Bias controlling length of demuxed pulses sent to the individual current mirrors of each pixel
AER_IN_PD	76	rec_gain-	Supply for left pmos of the current mirror of each pixel in the AER receiver.
AER_SERIAL_IN	77	clk!	Clock

Table 2 Pin list. All pins relevant for making the whole visual disparity computing system work are listed. Both the names used in the layout and schematics of the chip are liated for reference. The description gives a quick overview of the purpose/task of the pin in question

PIN NAME IN LAYOUT	NR	PIN NAME IN SCHEMATIC	DESCRIPTION
ANALOG<2>	4	Iin2_Mirror_WTA_test	Controls amount of current flowing out of the second current mirror in the CM/WTA circuit. (Gate of NMOS transistor.)
CONTROL<0>	5	Iin1_Mirror_WTA_test	Controls amount of current flowing out of the first current mirror in the CM/WTA circuit. (Gate of NMOS transistor.)
DATA<4>	11	Iout2_Mirror_WTA_test	Output from line 2 of CM/WTA test circuit.
DATA<5>	12	Iout1_Mirror_WTA_test	Output from line 1 of CM/WTA test circuit.
DATA<10>	17	Vout_on_test	Vout of output neuron.
DATA<11>	18	Iout_CD_test	Current output of the test copy of a coincidence detector.
DATA<12>	19	Vout_in_test	Vout inputneuron test
ANALOG<3>	46	Vpulselength2	Same function as Vpulselength but for the test copy of the input neurons.
ANALOG<5>	48	Iin_on_test	Current input for test copy output neuron.
ANALOG<12>	59	Vin2_CD_test	Voltage input 2 for the test copy of a coincidence detector.
ANALOG<13>	60	Vin1_CD_test	Voltage input 1 for the test copy of a coincidence detector.
CONTROL<5>	61	Iin_in_test	Gate of a PMOS transistor used as a current source for the test copy of the input neuron.

Table 3 Pin list. Only pins for the extra test circuits of the chip are listed. Both the names used in the layout and schematics of the chip are listed for reference. The description gives a quick overview of the purpose/task of the pin in question.