

# CRYPTO SYSTEMS

Team: Philippe Achkar, Michael Gradek, Frederic Thouin

Title: Simple and Symmetric: the **RC4 Encryption Algorithm**

Abstract: Since the advent of the Internet and of network transactions, [...] protect sensitive information. The objective is to find a simple, fast, reliable and easy to implement solution which is strong enough for security needs. The RC4 encryption algorithm proves to be an ideal choice for those wishing to ally strength, simplicity and efficiency. RC4 is a symmetrical algorithm, which means the same method, or key, is used for encryption and decryption; thus, the key does not need to be transmitted with the message. Furthermore, RC4 is a strong algorithm: each key can only be used once, increasing the level of protection without making the algorithm more complex, while only a small set of keys are considered weak. Finally, RC4 is ten times faster than the DES algorithm. These advantages make the RC4 algorithm interesting to study and implement, and this is what we propose to do here.

In this project, we will develop an RC4 encryptor and decryptor on an Altera FPGA chip, using VHDL. We intend to illustrate the inner workings of the RC4 mechanism, by streaming in unencrypted bits, encrypting them using a key and showing the encrypted result at the output. This output will then be decrypted using the same key using a distinct module. We will build scalable n bit models which minimize latency; these modules can be combined to form larger modules, which would then maximize throughput.

[1] M. Shaffer, "RC4 Encryption Using ASP & VBScript," January 2000,

[2] C. Grogans, J. Bethea , I. Hamdan , North Carolina Agricultural and Technical State University, "RC4 Encryption Algorithm," March 2000,

[3] A. Roos "Weak Keys in RC4," Posted on newsgroup sci.crypt, 22 Sep 1995.

[4] R.J. Jenkins Jr., "ISAAC and RC4," 1996

Team: Moayad Fahim Ali, Mohammad Sharif Al-Saati

Title: Hardware implementation of the **Data Encryption Standard** algorithm

Cryptography is the process of maintaining and communicating in secret writings or ciphers, which is essential nowadays for organizations to maintain privacy and data security. Several cryptography algorithms exist, with the more advanced ones requiring the use of a "key" to control the encryption and decryption processes. These algorithms can be grouped into Public (asymmetric), or Secret (symmetric) key algorithms. Asymmetric algorithms make use of different keys for encryption and decryption as opposed to symmetric algorithms which use the same key for both processes.

The Data Encryption Standard (DES) symmetric algorithm is suitable for larger messages because it is a fast block-encryption cipher; it encrypts entire data blocks at a single time. The proposed project implementation will allow for 64-bit data block encryption using a 56-bit randomly generated key. This means that the data blocks, which are operated on as bits, can be transformed into  $2^{64}$  possible combinations. A key used for a specific application can only generate unique encryption of the data; hence unauthorized recipients cannot recover the ciphertext without the key. This project design involves the implementation of the DES algorithm using VHDL. The design will be tested and simulated on the Altera MaxPLUS II software.

#### References:

- [1] D. Stinson, ?Cryptography: Theory and Practice?. CRC Press, 1996.
- [2] R. A. Mollin, ?An introduction to cryptography?. Chapman & Hall/CRC, 2001.
- [3] D. E. Denning, ?Cryptography and data security?. Addison-Wesley, 1982.
- [4] J. O. Grabbe, ?The DES Algorithm Illustrated?

Team: Mark Belcarz and Franco Puccio

Title: Design of the **RSA public-key cryptosystem**

Cryptography, simply defined, is the process of combining some input data, called plaintext, with a user-specified password, called a key, to generate an encrypted output, called ciphertext, in such a way that, given the ciphertext, no one can recover the original plaintext without the encryption password in a reasonable amount of time.

This project shall concentrate on the design a encryption/decryption system based on the RSA algorithm (named after Ron Rivest, Adi Shamir and Len Adleman who invented it in 1977) using VHDL. The goal is to be able to encrypt and decrypt a message being sent between two communicating parties, using both a public and private key. The basic algorithm is illustrated in the attached file.

### RSA Algorithm

<b>Key Generation</b>	<b>Encryption</b>
1. Generate two large prime numbers, $p$ and $q$	$C = P^e \% n$
2. Let $n = pq$	<b>Decryption</b>
3. Let $m = (p-1)(q-1)$	$P = C^d \% n$
4. Choose a small number $e$ , coprime to $m$	
5. Find $d$ , such that $de \% m = 1$	
Publish $e$ and $n$ as the public key. Keep $d$ and $n$ as the secret key.	$x \% y$ means the remainder of $x$ divided by $y$

Delfs, Hans, "Introduction to cryptography: Principles and applications" Springer, 2002.

Stinson, Douglas R, "Cryptography: Theory and practice" Chapman & Hall, 2002.

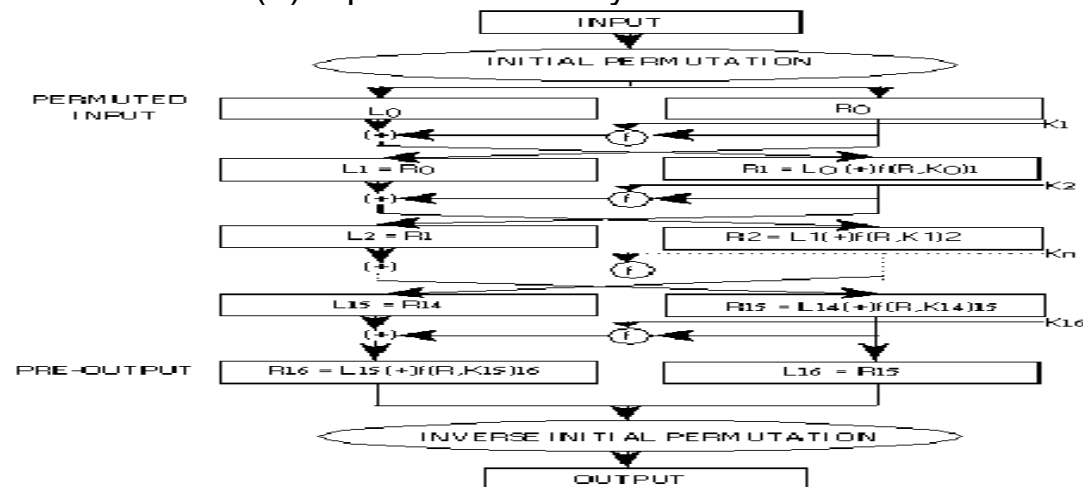
Graham, Ian G, "A Brief Introduction to Cryptography",

Team: Steve Crisafulli and Stylianos Giannelis  
 Title: **Data Encryption Standard (DES)** algorithm

The algorithm will be designed to encrypt and decrypt blocks of data consisting of 64 bits under control of a 64-bit key. Decrypting will be accomplished by using the same key as for encrypting, with the exception that it will be accessed in the reverse order. A block to be encrypted will first be subjected to an initial permutation (IP). Subsequently, it will iterate 16 times through a complex key-dependent computation and finally it will be subjected to an inverse permutation (IP-1). The IP and IP-1 units will alter the order of their inputs. The key-dependent computation will be simply defined in terms of a function  $f$ , called the cipher function, and a function  $KS$ , called the key schedule. The input block will be split into two blocks  $L$  and  $R$  where  $L$  corresponds to the left half of the input block and  $R$  to the right half. At each of the 16 iterations, a new  $K$ ,  $L$  and  $R$  will be computed according to the following formulas:

$$K_n = KS(n, KEY), L_n = R_{n-1} \text{ and } R_n = L_{n-1} (+) f(R_{n-1}, K_n)$$

where  $n$  represents the iteration and  $(+)$  represents a bit by bit addition modulo 2.



[1] B. Schneier, "Applied Cryptography", 2nd Edition, Wiley 1996.  
 [2] J. Gilmore, "Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design", May 1998.  
 [3] R. K. Nichols, "ICSA Guide to Cryptography", McGraw-Hill, 1999.  
 [4] Federal Information Processing Standards Publications, "Announcing the Standard for Data Encryption Standard", <http://www.itl.nist.gov/fipspubs/fip46-2.htm>.  
 [5] Tropical Software, "DES Encryption", <http://www.tropsoft.com/strongenc/des.htm>.

Team: Geoff Mayhew & Robert Litvack

Title: Implementation of **RC6 Block Cipher Cryptography System**

In the ever-evolving world of data security, technological methods are constantly being improved upon to better protect information. Encryption and decryption ciphers are powerful tools in the field of data protection. While many different algorithms exist, we have chosen to investigate and realize the RC6 algorithm.

Designed by Ron Rivest in collaboration with associates from RSA laboratories, the RC6 block cipher is an evolutionary improvement of RC5. It was designed to meet the requirements of the Advanced Encryption Standard (AES). In order to be AES compliant, a block cipher must utilize 128-bit input/output blocks. The RC6 cipher employs four 32-bit registers for operations that have the advantages of doing two rotations per round and using more bits of data to determine rotation amounts in each round. Also, it includes integer multiplication as an additional primitive operation, a feature not found in RC5. Overall, RC6 maximizes the diffusion achieved per round, allowing for greater security, fewer rounds and increased throughput.

Therefore, we will implement the RC6 cryptosystem algorithm in VHDL intended for use on the Altera Flex 10k series FPGA.

#### References:

1. RSA Laboratories | RC6 Block Cipher <http://www.rsasecurity.com/rsalabs/rc6/>
2. Ronald L. Rivest, M.J.B. Robshaw, R. Sidney, Y.L. Yin, "The RC6 --- Block Cipher" <ftp://ftp.rsasecurity.com/pub/rsalabs/rc6/rc6v11.pdf>
3. "The RC6 Block Cipher: A simple fast secure AES proposal" <http://csrc.nist.gov/encryption/aes/round1/conf1/rc6-slides.pdf>

Team: Anh-Quang Nguyen, Allison Smedley

Title: A Hardware Implementation of the **International Data Encryption Algorithm (IDEA)**

Data encryption has been an increasingly important concept in the development of secure communications. There is a vast array of different algorithms that exist to encrypt and decrypt data. IDEA (International Data Encryption Algorithm) was developed at ETH (the Swiss Federal Institute of Technology) and patented by the Swiss firm Ascon. Compared to the DES (Data Encryption Standard) algorithm, IDEA has been shown to be faster and more secure.

For IDEA, the same algorithm is used for encryption and decryption. It is a block cipher algorithm that operates on 64-bit plaintext blocks, using a 128-bit key. The 64-bit input block is divided into four 16-bit blocks, which become input blocks to the first round of the algorithm. There are eight rounds in total, and in each round the four blocks are XORed, added, and multiplied with subkeys based on the original key. Between each round, the second and third subblocks are swapped. The algorithm design will be coded in VHDL and implemented on an Altera Flex10K FPGA. In order to implement the algorithm within time and hardware constraints, the block size will be 16 bits rather than 64 bits.

#### References:

- [1] Crypto Systems Incorporated (2002). [Online]. The IDEA Encryption Algorithm. Available: <http://www.finecrypt.net/idea.html>. February 17, 2003 (accessed).
- [2] Frame Technology (1994). [Online]. International Data Encryption Algorithm (IDEA). Available: <http://www.cs.nps.navy.mil/curricula/tracks/security>. February 17, 2003 (accessed).
- [3] A. J. Menezes. Handbook of Applied Cryptography. Boca Raton: CRC Press, 1997.
- [4] J. G. Savard (2000). [Online]. IDEA (International Data Encryption Algorithm). Available: <http://home.ecn.ab.ca/~jsavard/crypto/co040302.htm>. February 17, 2003 (accessed).
- [5] B. Schneier. Applied cryptography : protocols, algorithms, and source code in C. New York: Wiley, 1996.
- [6] TechTarget (2003). [Online]. International Data Encryption Algorithm. Available: <http://searchsecurity.techtarget.com>. February 17, 2003 (accessed).

Team: Jean-Sebastien Siow and Celine Ho

Title: **RSA** (Rivest, Shamir, and Adelman) **public-key Cryptosystem**.

RSA is a popular algorithm used for encrypting and decrypting data. The algorithm was developed by three mathematicians Rivest, Shamir, and Adleman. The difficulty in cracking RSA encryption is the reason why RSA is implemented in many communication and security signatures today. There is an increased need for secure transmission of data (online banking, intranet communication, digital cellular phones) and a reliable means to encrypt data is required.

Developed in 1977, the system uses a private and public key. Every user has a pair of keys: a public key, which is provided to all partners and a private key. A message is encrypted with a public key and can only be decrypted with the private key and vice-versa. The security of RSA is based on the difficulty that lies in factoring large numbers as opposed to multiplying two large numbers.

The same hardware can be used for encryption as well as decryption. For our project, we will implement an RSA encryption/decryption system using VHDL.

1. T. Cormen, C. Leiserson, R. Rivest, Introduction to Algorithms, McGraw Hill Publishers, 1991.
2. Hennessy, J.L., Patterson, D.A., Computer architecture: a quantitative approach, Morgan Kaufman Publishers, Inc., 1990.
3. T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on Information Theory, 31(4):469-472, July 1985.
4. R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM, vol. 21, pp.120-126, 1978.
5. Cetin Kaya Koc, RSA Hardware Implementation, RSA Laboratories, Version 1.0, 1995.

Team: Jason Tea, Hong Bae Kim

Title: FPGA Implementation of the **RSA Cryptography Algorithm**

Cryptography is a strong security measure to protect information, maintain privacy and confidentiality. As we move into an information society, cryptography has become one of the main tools for access control, electronic payments, corporate security on the internet and various types of networks.

Many cryptography algorithms are currently available but RSA algorithm has become a standard for industrial-strength encryption, especially for data sent over the internet. The RSA algorithm was named after Ron Rivest, Adi Shamir and Len Adleman, who invented it in 1977. The basic ingredient in the RSA cryptosystem algorithm is its computational complexity.

Our project presents the mathematical analysis of the RSA algorithm, where the public key and private key are used to encrypt and decrypt messages. Also, the analyzed RSA algorithm will be designed in VHDL and synthesized for implementation on a MAX+PLUS II Altera's FPGA chip.

- [1] Denning, Dorothy E., "Cryptography and data security", Addison-Wesley Publishing Company, Inc., 1982.
- [2] Jennifer Seberry and Josed Pieprzyk, "Cryptography: An Introduction to Computer Security.", Prentice-Hall, 1989.
- [3] C. H. Meyer and S. M. Matyas, "Cryptography: A new dimension in computer data security", John Wiley & Sons Inc., 1982.
- [4] <http://www.ssh.com/support/cryptography>
- [5] <http://oregonstate.edu/dept/honors/makmur/>
- [6] [http://www.di-mgt.com.au/rsa\\_alg.html](http://www.di-mgt.com.au/rsa_alg.html)
- [7] <http://www.rsasecurity.com/>

Team: Chen Sheng and Dominic Vallelonga

Title: FPGA Implementation of the **RSA Encryption Algorithm**

With today's rapidly evolving communications market the need for secure transmission of data is of critical importance. The RSA encryption algorithm (Rivest, Adi Shamir, Adleman), invented in 1977, is used in most public-key cryptography systems. Its security is based on the difficulty of factoring large numbers. In this project, a VHDL implementation of the RSA cryptographic system is presented based on the modular multiplication proposed by P. L. Montgomery.

RSA is an asymmetric cryptosystem that uses one key (public key) to encrypt a message and a different key (private key) to decrypt it; therefore, both encryption and decryption circuits will be implemented. In order to optimize the speed of the circuits, the RSA encryption algorithm employed should be able to carry out modular exponentiation efficiently. For this purpose, the Montgomery multiplication will be used to perform the modular multiplication and squaring during the exponentiation process.

- [1] A. J. Menezes, P. van Oorschot, and S. Vanstone, "Handbook of Applied Cryptology.", CRC Press, 1996
- [2] Bruce Schneier, "Applied Cryptography." Second Edition, John Wiley & Sons, 1996.
- [3] M. Shand and J. Vuillemin, "Fast Implementations of RSA Cryptography.", Proc. 11th IEEE Symposium on Computer Arithmetic, Canada, July 1993.
- [4] Richard Holowczak, "RSA Demo Applet.",  
<http://cisnet.baruch.cuny.edu/holowczak/classes/9444/rsademo/#overview>
- [5] RSA Laboratories, "PKCS #1 v2.1: RSA Encryption Standard.",  
<http://www.rsasecurity.com/rsalabs/pkcs/pkcs-1/index.html>

# COMPUTATIONAL GRAPHICS

Team: Adil Ahmad and Justin Rizk

Title: **Anti-Aliasing** for Computer Generated **Images**

Anti-Aliasing is a very important part of generating high quality 3D images. There are various approaches to perform this task. The most common approach is a supersampling approach, where the image is generated at 2 or 4 times its resolution, and average the intensity of each pixel that is actually drawn. The main problem with this approach is that it is computationally very expensive.

The goal of our project is to implement an anti-aliasing which is much less computationally expensive. Our approach will be one which does not use supersampling, at least not on the whole scene. By detecting where the edges (jaggies) occur, we will calculate some weighting ratio to smooth out the jaggies. This approach is simple, but much less expensive than the supersampling approach mentioned above, thus making it an ideal candidate for real-time graphics processing.

References:

1. Tsuyoshi Isshiki & Hiroaki Kunieda, "Efficient Anti-Aliasing Algorithm for Computer Generated Images"
2. Yuan-Hau Yeh & Chen-Yi Lee, "A New Anti-Aliasing Algorithm for Computer Graphics Images"
3. Foley, van Dam, Feiner, Hughes, "Computer Graphics: Principles and Practice" 2nd edition in C, Addison-Wesley, July 1997

Team: Genevieve Bertrand and Connie Triassi

Title: Hardware Implementation of the **2D Graphics Pipeline for Polygon Rendering**

The performance of most graphics system is largely dependant on the speed at which it can render primitives, such as circles and polygons. To be completely rendered, primitives have to go through the 2D graphics pipeline, which consists of three conceptual stages: the application program, which feeds commands to the CPU, the geometry subsystem, and the rasterization subsystem. Although all three parts may be implemented in software, the amount of calculation involved in rendering a primitive is very significant, and since the number of primitives in one image is typically very large, it is usually beneficial to accelerate parts of the pipeline in hardware.

Thus, this project will focus on the hardware implementation of the geometry and rasterization subsystems of the graphics pipeline for polygon rendering. More specifically, we will consider triangle primitives because they are the most often used polygons in graphics. The geometry stage will perform per-vertex operations such as geometric transformations (rotations, translations) and clipping. The latter will be done using the Sutherland-Hodgeman clipping algorithm, which is easily implementable in hardware. The rasterization component, on the other hand, will work on individual pixels and its main function will be scan conversion. Since the polygons considered here are formed of three lines, they will be rasterized with the Bresenham's line algorithm.

Finally, the main goal of this project will be to optimize throughput, i.e. to render as many polygons per second as possible. This objective will be achieved by further pipelining each component of the graphics system, which is itself already pipelined.

- [1] D. Hearn and M.P. Baker, "Computer Graphics, C Version", New Jersey: Prentice Hall, 1997.
- [2] J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes, "Computer Graphics: Principles and Practice in C", second edition, Addison-Wesley, 1996.
- [3] E. Angel and D. Morrison, "Speeding Up Bresenham's Algorithm", IEEE Computer Graphics and Applications, vol. 11, no. 6, pp. 16-17,
- [4] M. Olano and T. Greer, "Triangle Scan Conversion using 2D Homogeneous Coordinates", presented at the 1997 SIGGRAPH / Eurographics Workshop on Graphics Hardware,
- [5] F. Pfenning, "Clipping and Scan Conversion", class notes for 15-462 Computer Graphics I, School of Computer Science, Carnegie Mellon University in Pittsburgh PA, March 2002

Team: Antoine Lourtau and Gregory Zoughbi

Title: Single-Color **Texture-Mapping Unit** with MIP-mapping and Bilinear Interpolation.

Texture mapping is a fundamental technique for rendering high quality images inexpensively into a frame buffer by mapping an image, or a texture, to a polygon. Because different dimensions of textures and polygons may cause "sparkling", "moiré" or "blocky" artifacts to appear, a texture mapping unit uses a "Texture Magnification Filter" when the texture is considerably smaller than the polygon and a "Texture Minification Filter" when a texture is considerably larger than the polygon.

This paper describes a texture mapping unit that uses bilinear interpolation for texture magnification and the "Nearest MipMap Nearest" MIP-mapping algorithm for texture minification. This 2-D unit supports a frame buffer resolution of 256x256 pixels, a texture resolution up to 256x256 texels, and a single 8-bit color channel. It can be cascaded with other chips to generate a 3-color system, and can be arranged in parallel with other chips to support a larger frame buffer resolution.

This design attempts to maximize throughput at the expense of resources; a given texture is immediately filtered and its corresponding MIP-maps are generated in order to optimize its mapping time into polygons.

#### References:

[1] D. Hearn and M. P. Baker, "Computer Graphics, C Version", 2nd edition, Prentice-Hall, 1994.

[2] P. Heckbert, "Fundamentals of Texture Mapping and Image Warping", 1989,  
[http://www-users.itlabs.umn.edu/classes/Fall-2001/csci5980/notes/Texture\\_Mapping1.ppt#1](http://www-users.itlabs.umn.edu/classes/Fall-2001/csci5980/notes/Texture_Mapping1.ppt#1).

[3] M. Woo, J. Neider, and T. Davis, "OpenGL Programming Guide", 2nd edition, Addison-Wesley, 1998.

[4] P. Haeberli and M. Segal, Silicon Graphics Inc. (SGI), "Texture Mapping as a Fundamental Drawing Primitive", 1993, <http://www.sgi.com/grafica/texmap/>.

Team: Jeff Singer, Elliot Sinyor

Title: **2D Antialiased Line-Drawing Unit**

A fundamental operation on any raster graphics system is line drawing. Due to the inherent limitations in the resolution of the display device, lines tend to have a jagged or stair-step appearance. This distortion of information due to low-frequency sampling is known as aliasing. This attracts unwanted attention in both static and dynamic graphics. In the case of a moving image, aliased lines flicker and have crawling edges. Anti-aliasing seeks to compensate for this undersampling by varying the intensity of the border pixels to reduce the jagged effect.

We are implementing a line-drawing unit that uses the Bresenham algorithm to draw the line, and Wu's algorithm for anti-aliasing. The inputs to the unit will be the endpoints of the line, and the output will be the coordinates of the pixels as well as their intensity. The unit will be described in VHDL and synthesized on an FPGA.

References:

1. Hearn, Donald - Computer graphics, C version (2nd Ed.), Upper Saddle River, N.J. : Prentice Hall, 1997.
2. Thaddeus, Jude - "Wu's Anti-aliasing Line Drawing" <http://www.sanctuary.org/~azimuth/coding/aaline.html>
3. Elias, Hugo - "Wu-Anti-aliased Lines" [http://freespace.virgin.net/hugo.elias/graphics/x\\_wuline.htm](http://freespace.virgin.net/hugo.elias/graphics/x_wuline.htm)

# IMAGE PROCESSING

Team: Feras AbuTalib and Hisham Shafiq

Title: A VHDL implementation of the **MultiScale Retinex with Color Restoration Algorithm**

Subtle differences exist between the direct human observation of scenes and that of recorded color images. One of the big differences is the fact that the human visual system is able to distinguish details and vivid colors in shadows and in scenes that contain illuminant shifts. A number of algorithms have been proposed to bridge this gap. The Multiscale Retinex with Color Restoration (MSRCR) has shown itself to be one of the best algorithms in this field. This project will attempt to implement the MSRCR in hardware. This will allow photographers to get automatic image enhancement directly from their digital cameras. The algorithm can be summarized by

$$\hat{R}_{MSRCRi}(x, y) = G \left[ C_i(x, y) \sum_{n=1}^N (\log I_i(x, y) - \log [I_i(x, y) * F_n(x, y)]) \right]$$

where  $I_i(x, y)$  denotes the image distribution in the  $i$ 'th color spectral band,  $*$  the convolution operation,  $F_n(x, y)$  the surround function,  $N$  the number of scales,  $G$  is the weight associated with each scale, and  $C_i(x, y)$  is given by

$$C_i(x, y) = \log[\alpha I_i(x, y)] - \log \left[ \sum_{i=1}^S I_i(x, y) \right]$$

where  $\alpha$  is the color restoration constant and  $S=3$  for images in the RGB color scheme.

There are mainly two goals in this project; the first one is to achieve good approximation of this algorithm in hardware, whereas the second is to examine the performance of our hardware approach and compare it to the software based approaches. Simulations of an FPGA programmed using VHDL will be presented and compared to its matlab counterpart.

1 Jobson, D.J.; Rahman, Z.; Woodell, G.A.; Properties and performance of a center/surround retinex Image Processing,

2 Jobson, D.J.; Rahman, Z.; Woodell, G.A.; A multiscale retinex for bridging the gap between color images and the human observation of scenes Image Processing

3 Rahman, Z.; Jobson, D.J.; Woodell, G.A.; Multi-scale retinex for color image enhancement Image Processing, 1996.

4 Marina Balabanov and Yaron Zalika ; Image enhancement method based on Multiscale Retinex

5 A. K. Jain, Fundamentals of Digital Image Processing.

Team: Hari Angepat, Jonathan Campeau

Title: **Maximal Throughput** Implementation of the **Sobel Edge Detection Algorithm**

Image processing and recognition techniques have garnered a remarkable amount of interest over the past few years. Due to the complexity in such analysis, feature detection and extraction is the first step in most image segmentation techniques. One such method of accomplishing this extraction is thru edge detection. The primary goal of this process is to develop sets of closed pixel contours surrounding regions of interest in the image.

The Sobel edge algorithm uses two convolution kernels to detect intensity differences along the x and y axis. The result of the operation can be represented as a 2D spatial gradient. This allows for the emphasis of high spatial frequency areas corresponding the edges of an image.

As the processing of a video stream has substantial bandwidth requirements, the hardware implementation of the Sobel algorithm is quite useful in embedded and intelligent camera systems. This project will strive for a maximal throughput implementation, to allow for maximum frame rates and resolutions. Thus, the processing can occur in real-time without dropping any frames.

The Sobel Edge Detector uses a simple *convolution kernel* to create a series of *gradient magnitudes*. Applying convolution  $K$  to pixel group  $p$  can be represented as:

$$N(x, y) = \sum_{k=-1}^1 \sum_{j=-1}^1 K(j, k) p(x - j, y - j)$$

The Sobel Edge Detector uses two such convolution kernels, one to detect changes in vertical contrast ( $h_x$ ) and another to detect horizontal contrast ( $h_y$ ).

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

1. R.C. Gonzalez and R.E. Woods, "Digital Image Processing". 2nd Edition, Addison-Wesley, 1992.
2. Russ, John C. The Image Processing Handbook. CRC Press, Boca Raton, Florida, 1995.
3. James Matthews, "An Introduction to Edge Detection: The Sobel Edge Detector", 2002
4. R. Fisher, S. Perkins, A. Walker and E. Wolfar, "Sobel Edge Detector" 2000

Team: Carmen Au and Samir Diwan

Title: **2D Discrete Cosine Transform (DCT) for JPEG Image Compression**

The Discrete Cosine Transform (DCT) is currently the most widely used transform in image compression. Although there are other steps required in image compression, the majority of the processing required to encode or decode the images is done by the calculations performed in taking the DCT. The Discrete Cosine Transform is similar to the Discrete Fourier Transform however, it is much simpler because it avoids the use of complex numbers. The work it performs is the main bulk of the work done in image compression. In order to implement the DCT, the mathematical equation

$$y(u) = \frac{C(u)}{2} \sum_{i=0}^7 x(i) \cos\left(\frac{(2i+1)u\pi}{16}\right)$$

must be coded in VHDL. The 2D equation is overly convoluted for VHDL coding, thus as a simplification would be preferred. First it can be observed that the DCT is separable into 2 one-dimensional DCTs; one along the rows and one along the columns. Thus, in order to switch between rows and columns a transposing buffer will be required. This buffer consists of an 8x8 memory array for transposing matrices of partial products (see figure 1 in PDF attachment).



D.S. Taubman and M.W. Marcellin, JPEG2000 : image compression fundamentals, standards, and practice. Boston : Kluwer Academic Publishers, 2002.

P. Symes, Video compression demystified. New York : McGraw-Hill, 2001.

Pipelined Fast 2-D DCT Architecture for JPEG Image Compression.(September 2001)[Online]. 2003

The Cameron Project: High-Level Programming of Image Processing Applications on Reconfigurable Computing Machines1. (October 1998)

R. C. Gonzalez, R. E. Woods, Digital image processing. Upper Saddle River, N.J. : Prentice Hall, c2002.

Team: Irene Giannoumis and Malek Tabbara

Title: Implementation and design of an **EREC algorithm**

Most compression algorithms used today comprise some form of variable length coding (VLC). The basic idea behind VLC is to represent frequent data symbols by short bursts of binary digits, while data symbols that are less often encountered are represented by longer bursts of bits, resulting in a smaller average number of bits needed to represent all the symbols from the source.

One of the main drawbacks of VLCs is that they are highly sensitive to error propagation. In order to be able to reconstruct a sequence of symbols from a sequence of bits, the receiver requires all bits to be transmitted correctly. The aim of this project is to implement and design an algorithm for adapting VLC schemes to give increased resilience to random and burst errors while maintaining suitable compression performance.

Error resiliency will be achieved through the application of an EREC scheme that consists in multiplexing variable length codes into fixed length data blocks. The latter blocks are composed of slots, wherein each slot corresponds to the beginning of a VLC, thereby allowing the decoder to automatically synchronize at the beginning of each slot. The combination of error resiliency and data compression makes the scheme appropriate for data streams requiring high transfer rates.

The scheme will be tested under different noise conditions, to assess the trade-off between its compression performance and error-resilience characteristics, in the context of an image compression application.

Redmill, D.W.; Kingsbury, N.G.: The EREC: an error-resilient technique for coding variable-length blocks of data. IEEE Transactions on Image Processing, Volume: 5 Issue: 4, April 1996, pp. 565-574

Takishima, Y.; Wada, M.; Murakami, H : Variable length codes. IEEE Transactions on Communications, Volume: 43 Issue: 2 Part: 3 , Feb.-March-April 1995 pp. 158-162

Team: Mayank Mehta

Title: **Character Recognition** through **Neural Networks**

This project seeks to apply neural networks in recognizing handwritten characters in the English language. A multi-layer Neural Network equipped with a back-propagation learning algorithm is used to enable fast learning. This allows for specific types of handwriting to be recognized faster over time (learning through the training of the neurons), thus allowing the program to be user-specific.

The input layer of the MLP (Multi-Layer Perceptron) is a set of image pixels. An X by Y pixel image is mapped onto X by Y neurons. During the training period, the neurons are recursively fed characters to build their fault-tolerance (variance between two different As). The training is done with no pre-programming or bias.

Using windowing (breaking-up the characters into four/six/eight equal pieces) expedites the recognition process. For example, if a set of pixels comprising the top left hand corner of the character matches the data in the trained neurons, the code need not do similar computations for the rest of the character.

The aim of using Neural Networks coupled with VHDL is to automate and accelerate the process of character recognition.

References:

1. E. W. Brown, "Character Recognition by Feature Point Extraction", unpublished paper written at Northeastern University, 1992
2. D. S. Levine; Introduction to Neural & Cognitive Modeling, 1991, Lawrence Erlbaum Associates, Publishers; Hillsdale, NJ
3. Patrick K. Simpson; Artificial Neural Systems, 1990, Pergamon Press, New York, NY
4. John Wester Introduction to Neural Networks  
<http://www.nd.com/neurosolutions/products/ns/whatisNN.html>

# COMPUTATIONAL CORES

Team: Alexandre Bouffard AND Patricia Pinard

Title: **Parallel String Matching** with Broadcasting Scheme

String matching is at the core of many modern computationally intensive activities such as DNA identification, internet search and database operation. Generally, string matching is performed at the software level but critical or large volume applications require a dedicated special purpose solution. On a single-CPU architecture the processor must load and then compare the pattern string piece by piece with the reference, and the task is even more complicated when some mismatches are tolerated. While the software approach is limited by its serial nature, the main benefit of an hardware implementation resides in its parallelism. In fact many comparisons can be performed simultaneously on one or many reference strings, speeding up the process by as much as the parallelism degree (number of streams used).

In this paper, we present the VHDL design of a parallel string matching unit synthesized for the Altera family of FPGAs and addressing the exact matching and k-mismatches problems. Our approach is based on the method developed by Park and George [1], in which a broadcasting scheme with multiple streams is presented. This parallel scheme has two major advantages: (1) the number of multiple streams can be chosen to gain the maximum benefit of the parallelism and (2) the scheme can work on unknown sized reference strings. In addition, the broadcasting scheme solves the exact matching and the k-mismatches problems with a time complexity of  $O(n/d)$ , where  $n$  is the length of the reference string and  $d$  represents the number of different input streams used. This algorithm achieves high performance by exploiting explicit parallelism. We then develop a parameterized VHDL entity in which the degree of parallelism is controlled explicitly by a variable number of input (and output) streams.

1. Jin H. Park and K. M. George, "Parallel String Matching Algorithms Based on Dataflow," Proceedings of the 32nd Hawaii International Conference on System Sciences, pp. 1-10, 1999.
2. Hsi-C. Lee and Fikret Ercal, "RMESH Algorithms for String Matching," Proceedings of the third International Symposium on Parallel Architectures, Algorithms, and Networks, pp. 223-226, 1997.
3. Dan Gusfield, "Algorithms on strings, trees, and sequences : computer science and computational biology", Cambridge, Cambridge University Press, 534 p., 1997.

Team: Essam Nessim and Stuart Dack

Title: Implementation of a **modified version of MIPS**

A RISC (Reduced Instruction Set Computer) is a processor whose design is based on the rapid execution of a sequence of simple instructions rather than a large variety of complex ones. The central idea is that by speeding up the most common, simple instructions, one could afford to pay a penalty in the unusual case of a complex instruction and make a large net gain in performance.

There are several computer architectures based on this concept. Our project is going to be based on the MIPS (Millions of Instructions Per Second) architecture. Our goal is to implement a subset of MIPS using VHDL that deals with integer operations only (as opposed to floating point ones). This compact version will contain the most basic instructions of MIPS such as: load, store, add, subtract, branch, and set less than. Figure 1 shows the implementation of such an instruction set. In this pipelined architecture, each instruction will take about 4 or 5 clock cycles to compute.

References:

- [1] J. L. Hennessy & D. A. Patterson, "Appendix A ? Pipelining: Basic and Intermediate Concepts," in Computer Architecture: A Quantitative Approach, 3rd Edition. San Francisco: Morgan Kaufmann, 2003.
- [2] G. Kane & J. Heinrich, MIPS RISC Architecture, 2nd Edition. Upper Saddle River, NJ: Prentice Hall, 1992.
- [3] D. Sweetman, See MIPS Run. San Francisco: Morgan Kaufmann, 1999.
- [4] A. Hui & J. Zappulla, School of Computer Science, Monash University, Clayton, Australia. "MIPS Architecture Animation," February, 2003, <http://www.csse.monash.edu.au/packages/mips/>.

Team: Jatinder Gharoo and Montu Gupta

Title: VHDL Implementation of **Floating Point Division**

Floating point number representations allow us to use real numbers on a computer. Floating point numbers consist of a sign, exponent, mantissa, and base. Many applications use floating point division, hence there is a need to develop a floating point divider. The circuit is modeled using the Very High Speed Integrated Circuits Hardware Description Language (VHDL).

Floating Point Numbers IEEE Single Precision

Exponent-8 bits, Mantissa-23 bits, sign-1 bit = Total 32 bits

Most computers support single (32-bit) precision formats. Single precision format can express numbers from  $(-3.4 \times 10^{38}$  to  $3.4 \times 10^{38}$ ).

The basic algorithm to divide two floating point numbers we are following is:

- Divide divisor mantissa from the dividend mantissa
- Subtract the exponent of the divisor from the dividend
- Compute the sign of the result based on the sign of the two operands
- Normalize the result

The divide by zero case will be handled separately with IEEE standard to represent a zero will be followed. Other exceptions like NaN will be handled as specified in IEEE standard.

References:

1. David Goldberg, "Computer Arithmetic" (appendix H), Xerox Palo Alto Research Center, 2003

Team: Marouaune Izouggaghen and Nikhil Angra

Title: Verification, Synthesis and Implementation of the **CORDIC** VHDL Code for a FPGA Device

Microprocessors have traditionally used a software approach for implementing algorithms used in digital signal processing (DSP). This approach was primarily used because of the difficulty in mapping these algorithms into hardware. In addition, the software solution provided a cost effective and flexible method but suffered in terms of speed. With the arrival of reconfigurable logic computers, a high-speed hardware solution is possible for implementing these algorithms. The Coordinate Rotation Digital Compute (CORDIC) is a hardware efficient algorithm that can be mapped into FPGAs. It hence provides a more attractive alternative to the traditional software approach. The CORDIC Algorithm has found many applications such as high-end calculators, radars, robotics and DSP transforms.

In this project, our main objective is to realize a hardware implementation of the most popular trigonometric and transcendental functions which CORDIC algorithm is known for viz. Sine, Cosine, Arctangent and Square Root. The CORDIC algorithm uses a simple and efficient approach based on shift-addition techniques together with vector rotations. These techniques are implemented in VHDL and mapped into FPGAs. The performance of the design is then compared to a software alternative.

#### References:

- 1) J. Abruzzo, "Applicability of CORDIC Algorithm to Arithmetic Processing", IEEE, 1985.
- 2) R. Andraka, "A survey of CORDIC algorithms for FPGA based computers", ACM, Monterey, 1998.
- 3) Behroos, Parhami, "Computer arithmetic: algorithms and hardware design", Oxford University Press, 2000.
- 4) P. Jarvis, "Implementing CORDIC Algorithms", Dr. Dobb's Journal, Oct 1990.
- 5) J. Volder, "The CORDIC computing technique", IRE Trans. Computers, 1959.
- 6) <http://www.andraka.com/files/crdcsrvy.pdf>
- 7) <http://www.ee.byu.edu/ee/class/ee621/Lectures/L22.PDF>

Team: Ahmed Usman Khalid, Bhavin J. Shastri

Title: **Emulation of quantum circuit elements** and Grover's Search Algorithm using FPGA technology

This project involves construction of some key elements of quantum circuits namely the Hadamard gates, controlled NOT-gates (CNOT gate) and the Conditional Phase-Shift gates. Since the behavior of these quantum gates does not conform fully to conventional circuit behavior, we would emulate the quantum behavior using traditional subsystems synthesized on a FPGA. In order to measure the usefulness of our emulated quantum gates we would then emulate the Grover's Search algorithm using our gates.

The field of quantum computation and cryptography is at its elemental stage where scientists and engineers are researching hard to understand complicated quantum procedures and useful implementations of exciting quantum phenomena. The difficulty currently faced in simulating such experiments in software is that software simulations can not successfully simulate the level of parallelism involved in quantum computations. One solution to this problem is to emulate these quantum systems using non-Von Neumann based hardware technologies (such as FPGAs) which allow us to perform these tasks in hardware where parallelism of quantum systems can be more effectively emulated.

#### References:

1. V. V. Shende, A. K. Prasad, I. L. Markov and J. P. Hayes, 'Reversible Logic Circuit Synthesis', in Proc. ACM/IEEE Intl. Conf. Comp.-Aided Design, pp. 353-360, November 2002.
2. G. F. Viamontes, M. Rajagopalan, I. L. Markov and J. P. Hayes, 'Gate-level Simulation of Quantum Circuits', Proc. Asia and South-Pacific Design Automation Conf., pp. 295-301, January 2003.
3. Michael Nielsen, 'Quantum Computation and Information'

Team: Obaid Malik and Ammar Khan.

Title: **Built-In System-Test for an ALU**

The increased functionality of today's microelectronic circuits, along with faster, denser, and smaller packaging is making it increasingly difficult to access, test, and verify chip and system functionality using traditional external equipment and instruments. Often, the circuits being tested are more complex than the testing units themselves, and so a complementary testing paradigm is needed. Built-In Self-Test (BIST) is one such paradigm. BIST attempts to move as much of the tester functionality as possible onto the silicon. Embedding the tester equipment functionality into the semiconductor product reduces the burden on and complexity of external test equipment, and increases speed since on-chip access is faster.

Our BIST project involves the design and implementation of a Built-In Self Test for a Arithmetic and Logic Unit. The overall design consists of three major parts: ALU, BIST (key Components: linear feedback shift registers / multiple input signature registers ) and a register-based micro-controller. The BIST system will test the correct working of the ALU unit in an efficient and time conserving manner. We will be creating our own ALU in order to ease the simulation of faults. The output signatures of standard non-erroneous ALU will be compared to the ALU under test in order to gauge the correctness of its outputs.

#### REFERENCES:

Avra, L.J.; McCluskey, E.J. Synthesizing for scan dependence in built-in self-testable designs. Visited Feb 20, 2003.

Kiefer, G.; Wunderlich, H.-J. Using BIST control for pattern generation. Visited Feb 20, 2003.

Mohamed, Abdil Rashid. Built-In Self-Test (BIST) <http://www.ida.liu.se/~zebpe/teaching/test/lec12.pdf>. ; Visited Feb 20, 2003.

Team : Sophia Lalani, Elie Nassar

Title : **General Purpose Processor**

There is a broad range of engineering concepts that could be implemented on almost all electronic gadgets, home appliances and cars, which in most cases are avoided due to the cost and complexity of the hardware. Thus were planning to develop a simple processor that is fully synthesizable with the simplest instruction set possible, and the least cost to develop it.

This processor will have a control unit, an arithmetic logic unit (ALU), register unit, and a memory. The instruction set will be composed of load, store, add, subtract, shift, AND, OR, XOR, jump and halt.

Because the objective of this processor is not to be implemented for real time systems we wouldnt worry about its frequency but the challenge is to make it reliable and as small as possible. Also the advantage is that it consumes so little silicon; there will be plenty of room to add other peripherals.

[1] Patterson, David A., and Hennessy, John L., Computer Organization & Design - The Hardware/Software Interface, Second Edition, Morgan Kaufmann, San Francisco, 1998, ISBN 1-55860-428-6.

[2] Structured Computer organization, Andrew S. Tanenbaum  
Third edition 1990, Prentice-Hall, Inc. ISBN 0-13-852875-1

[3] [http://ciips.ee.uwa.edu.au/~morris/CA406/CA\\_ToC.html](http://ciips.ee.uwa.edu.au/~morris/CA406/CA_ToC.html)

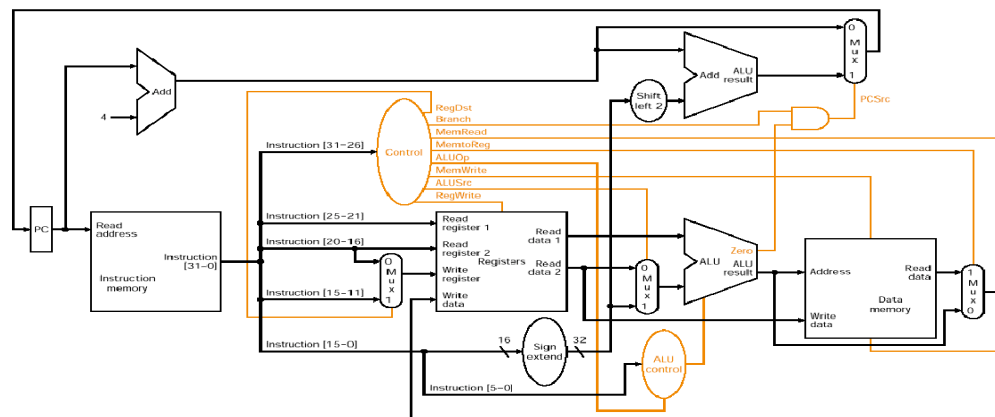
[4] <http://www.vautomation.com/Press/cd9707.html>

Team: Toma Sebastian Roseanu and Carla Hailpern

Title: A VHDL implementation of the **single cycle MIPS data path**

RISC processors are simple machines that can execute simple instructions at a high frequency. One of these processors is called MIPS. This project will implement a VHDL model of the single cycle MIPS datapath. This project can serve as a learning tool for beginner computer and electrical engineering students. The processor supports all basic operations including arithmetic, logical and branching instructions, which form the basis for all modern assembly code.

The aim of our project is to create a simple VHDL implementation of the single cycle MIPS processor datapath that can be independently used to simulate all MIPS core (non pseudo) instructions. We intend to implement the datapath by writing our own components in VHDL and using Altera LPMs to simulate data and instruction memory. Testing will be done by writing our own machine code which will subsequently be loaded into the instruction memory before the processor is started. We expect to have the results in the data memory. Our design will focus on simplicity and reliability to best suit our target audience.



1. John L. Hennessy, David A. Patterson, "Computer Organization and Design The Hardware/Software Interface" Morgan Kaufmann Publishers, 1998.
2. John. L. Hennessy, David A. Patterson, "Computer Architecture, A Quantitative Approach (3rd Edition)" Morgan Kaufmann, 2002.
3. <http://www-2.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15740-f97/public/doc/mips-isa.pdf> Title: MIPS IV Instruction Set, 1995 Author: Charles Price

Team: Hassan Najim and Youssef Karroum

Title: A 4-way **Set-Associative Cache Management Unit**

Cache is the name generally given to the first level of the memory hierarchy encountered once the address of a block of data leaves a CPU. It plays a very important role in computer performance, and deserves more attention and appreciation from computer users. In order to fully understand the functionality of a cache, one must examine the following four questions:

1. Where can a block be placed in a cache?
2. How is a block found if it is in the cache?
3. Which block should be replaced on a cache miss?
4. What happens on a write?

In our project, we will be implementing a 4-way set-associative cache management unit using VHDL. The four questions that were previously mentioned will be our design method ingredients. Our design would take, as inputs, block addresses (from the CPU) and give out, as outputs, control signals to the cache and to the lower level of memory hierarchy. This lower level of memory hierarchy is the random access memory, where the cache receives its blocks of data. In general these control signals will demonstrate how the CPU, cache, and RAM interface with each other.

Design methods for performance optimization will also be one of our main objectives. For instance, we will be using the write-back scheme with no-write-allocate as our design method of writing since we believe this will lead to better performance.

1. Patterson, Hennessy, Computer Organization And Design, The Hardware/Software Interface, Morgan Kaufmann.
2. Patterson, Hennessy, Computer Architecture: A Quantitative Approach, Morgan Kaufmann.
3. Hayes, Computer Architecture And Organization, McGraw Hill
4. Denis Howe, "The Free Online Dictionary of Computing",  
<http://foldoc.doc.ic.ac.uk/>.

# EMBEDDED PROCESSING

Team: Colicchio, Ettore and Crocker, Alana

Title: Electric Snowmobile **DC Brushless Motor PWM Controller**

The McGill Electric Snowmobile Team requires an electrically efficient controller for its DC brushless motor. The current controller has many features that are not necessary and lacks features that would be beneficial. Under the recommendation of Professor Ooi, we have chosen to design a motor controller that can adjust voltage levels depending on the throttle with multiple supporting features which will be tuned for direct use with the electric snowmobile. The design has to be energy efficient since the electric snowmobile needs to run in harsh conditions. Our current battery packs are regular lead-acid ones which do not perform well in the cold. We therefore need to keep the design small for minimal power consumption. Furthermore, since the team is fairly new, money is limited so smaller FPGA's would reduce the cost.

A detailed description of PWM is described in the attached PDF file (page 4-6). Since the snowmobile has a DC brushless motor, it requires a 3-phase AC signal to run. This AC signal needs to vary in amplitude to increase or decrease the torque output of the motor. However, the batteries used supply a "constant" DC voltage. This is where the controller comes in. The controller adjusts the constant DC voltage to a varying DC voltage depending on the throttle by switching two transistors (complements of each other) on and off for specific periods of time (PWM). We will also implement the following convenient features for the snowmobile specifically:

- Synchronous switching
- Regenerative
- Reverse mode
- Reverse and brake light control Cruise control with speed resume
- Extreme over-under voltage protection
- Dead-time tuning to support different transistor technology as well as 7 and 8-bit duty-cycle resolution

1. Adel S. Sedra, Kenneth C. Smith, "Microelectronic Circuits", Fourth Edition, Oxford University Press, 1998.
2. Bus Compatible Digital PWM Controller, IXDP 610  
<http://www.ixys.com/91600.pdf>
3. New Generation Motors. " NGM-EVC-200 series controller Operating Manual", Version 1.10D.

Team: Patrick Kechichian and Serge Koniski

Title: **A Missile Interception Model**

Missile interception is a defensive tactic which consists of launching a defensive blocking missile (the pursuer) in response to an enemy ballistic missile attack (the target). The pursuer's goal is to disable the target missile through a frontal collision in orbit. If the target manages to penetrate the atmosphere, there is nothing that can be done to stop it anymore. The tricky part in intercepting a missile is that the target is pre-programmed to change its trajectory during its flight. So the real pursuer is equipped with sensors that help detect any movement.

Our project will focus on a comparison between the hardware based approach (VHDL) and the software based approach (using C and/or Matlab). Each simulation will consist of a target launch followed by a pursuer launch. The progression of both entities will be traced and recorded. The final outcome will detect if the pursuer was able to make contact and intercept the target, or if the pursuer missed the target.

#### REFERENCES:

- 1- Zarchan, P. (1997) "Tactical and Strategic Missile Guidance" American Institute of Aeronautics and Astronautics
- 2- Ben-Asher, J. and Yaesh, I. (1998) "Advances in Missile Guidance Theory" American Institute of Aeronautics and Astronautics
- 3- Welch, G. and Bishop, G. (2002) "An introduction to the Kalman Filter" Department of Computer Science, University of North Carolina

Team: Prakash Patel, Tram Anh Nguyen

Title Testing and Implementation of **Convolutional Encoder and Viterbi Decoder** in VHDL.

With the advent of wireless communication, an increasing amount of data is being channeled through transmission lines. The occurrence of unwanted noise and interference is therefore a serious issue that must be dealt with since errors are being introduced into the transmission. One way to solve this problem is using an error correcting code scheme such as the convolutional encoder and Viterbi decoder.

The Viterbi decoder logically explores in parallel every possible data sequence. It compares each data component with different possible paths and picks up the closest match. For the implementation of the encoder/decoder in VHDL, the specification will be a 2 bit input, 1 bit output and memory order of 3.

The secondary goal is to create a test bench to measure the ability of the convolutional encoder and Viterbi decoder to fix errors in the signal. It is necessary to test extensively this coding scheme so that its performance can be compared to simulations done in C++ and MATLAB. Finally, the noise and interference will be simulated on the FPGA chip.

[1] Wang, Y et al. DSP Applications for Real Time Equalization, 2002. Note: Paper written for ECSE-494C.

[2] Garr, David A. Iterative Decoding for the GSM System, 1998.

[3] Huang, Fu-Hua. Chapter 3: Turbo Code Encoder, <http://scholar.lib.vt.edu/theses/available/etd-71897-15815/unrestricted/chap3.pdf>

[4] Sklar, Bernard. Digital Communications Fundamentals and Applications Second Edition. New Jersey: Prentice HALL, 2001.

Team: Christian Duchesneau and Anju Sharma

Title: VHDL Implementation of **Selective Repeat Error Recovery**

In today's rapid advancement and intensive usage of communications, the accuracy and efficiency of data transmission across networking channels is very significant. In order to achieve reliable data transfer between two end systems, an Automatic Repeat Request (ARQ) protocol must be used. Stop-And-Wait, Go-Back-N, and Selective Repeat are the three ARQ protocols in use today, the latter being the most efficient and complex one. These schemes make use of flow control, sequence numbers, acknowledgements, and timers to ensure that the data is delivered from a sending process to a receiving process correctly and in order.

Even though the Selective Repeat procedure is one of the most complex ARQ protocol, its 'renowned high channel efficiency' creates our motivation to implement it in VHDL. In this project, a simplified version of the Selective Repeat error recovery will be developed. Nevertheless, the simplifications will not alter the error recovery method used by this ARQ protocol, a method consisting in retransmitting the lost packets only. The SR error recovery operation will be observed through the simulation of a sender-receiver data transaction, and its performance will be evaluated by its protocol throughput and its rate of data delivery.

#### References:

- [1] J. F. Kurose, K. W. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, Addison-Wesley, 2nd edition, 2003
- [2] Srinidhi Varadarajan, Reliable Transmission: A State Machine Reliable Transmission  
[http://courses.cs.vt.edu/~cs5516/spring02/reliable\\_tx\\_1.pdf](http://courses.cs.vt.edu/~cs5516/spring02/reliable_tx_1.pdf)
- [3] Don Towsley, Chapter 3: Transport Layer  
<http://www.cs.umd.edu/~shankar/417-F01/Slides/chapter3a/sld033.htm>
- [4] <http://www.erg.abdn.ac.uk/users/gorry/course/arq-pages/sr.html>