



Nethra implementation of Swift: Primary Image-Processing for DNA sequencing, Using Massively-Parallel MIMD Silicon Computation

Nethra Engineering, MPPA Group
July 2010, Draft v0.91

This paper explores how the data processing components of DNA sequencing can be implemented using Nethra's massively-parallel, image-processing chips, with a focus on the initial image-processing phase. This preliminary study reflects the author's current understanding of the processes involved, when using the Swift approach as described in the Cambridge/Harvard paper as applied to Illumina's sequencing platform. Greater understanding and refinements will emerge over time.

Nethra's software-programmable image-processing silicon architecture is superior to the other commonly available massively-parallel silicon, specifically GPUs, in several fundamental ways:

1. Nethra's MIMD (multiple-instruction-multiple-data) silicon architecture enables easy-to-develop, computationally-efficient single-pass processing of an 'image-neighborhood context', whereas GPUs' SIMD/SIMT architecture must do atomic operations, then cache and re-order, requiring many passes over the same neighborhood data, with extensive external memory caching. This is a significant advantage for real-time processing or off-line throughput.
2. Nethra's instruction-set and ALUs are also biased toward image/video processing.
3. Due to the architecture advantage, combined with a straightforward programming model and mature software development tools, time-to-market can also be improved.
4. Nethra's chip is 15 times more energy-efficient, so it scales up or down much more easily with respect to form-factor, air-conditioning, and electricity operating costs. Whether it's a very large sequencer machine or a bench-top machine, it's smaller, and cheaper to develop and operate.
5. Nethra's medical, video and military embedded-systems customer base expects long-lived chip supply of 10-15 years, and Nethra has a proven 2nd source for the chip as well. Particular models of GPU's routinely reach end-of-life long before many military, video and medical systems do.

Nethra's Ambric-architecture™ Am2045™ processor has strong potential for DNA sequencing cost-reduction, throughput-increases, and has 10-15X better form-factor/power/heat advantages over CPUs and GPUs respectively.

SWIFT/Illumina Style Computation Analysis on Nethra Am2045:

The Swift package operates on a set of images representing one tile location from a flow-cell. Raw image data is collected and presented in the following form:

N cycles of
4 channels/tile
2048x1794x16bit images

Each cycle represents a sequential position along each of 1000s of DNA fragments, gathered into identical clusters.

Each channel represents one of the 4 bases: A, C, G, T. Images can be thought of as 16-bit monochromatic in their respective channel.

The Swift code has many run-time parameters. For the Nethra chip implementation, some will be used as build-time properties.

The major stages in DNA sequencing:

1. Image-Processing
2. Base Calling
3. Alignment

This draft of the application note only considers the Image-Processing stage, as it is completely separable from the Base Calling and Alignment stages and is a very good fit for Nethra's massively-parallel semiconductor technology. Image-processing is computationally very efficient on Nethra's Ambric-Architecture Am2045 Massively Parallel Processor Array (MPPA) chip with over 300 cores. The other stages will be analyzed at another time.

Because of the processing in step 1.2 below, there are 2 passes over the image data for a given tile. A host application will send the image dataset twice in succession. For small cycle counts it would be possible to store all images in local DDR, but it would not be general. Also, cycle counts are increasing over time.

Each image is approximately 7.5MB. The 4 images (called an 'image set') for a single cycle are approximately 30MB.

1. Image Processing

The purpose of the Image Processing step is to remove various artifacts from the tile images, so that the bases can be identified with high confidence.

The input to Image Processing is the complete set of images for all cycles, for the tile.

The output of Image Processing is, for each cluster location on the tile at each cycle, a list of the 4 base intensities.

1.1. Background subtraction

This occurs for each channel on every cycle.

Images are processed independently.

For each pixel, compute the minimum pixel value in a $M \times M$ -radius square neighborhood, and subtract that value from the input pixel. M is typically 6.

Estimates:

- Horizontal direction is fairly easy: est 10 clocks per pixel.
- Vertical direction requires a DDR connection, but still est 10 clocks per pixel.
- Original image is also stored in DDR, since it is needed after the minimum calculation is done.
- Clocks: est 20 clocks per pixel = 75Mclocks/image
- Processors: 7, SR and SRD mix
- RU Banks: 2 for small FIFOs
- DDR Connections: 1 (3 shared virtual connections)
- DDR Bandwidth: 50MB/image

1.2. Image correlation

Image correlation is performed in order to compensate for 2 sources of mis-alignment:

- Stage Offsets: Stage positioning between cycles.
- Inter-Channel Offsets: Optical frequency differences between channels.

Initial Computations:

For inter-channel offsets, the following is done once and the results recorded.

- For each channel, generate the threshold image for each cycle, and combine them together. The resulting 4 threshold images should contain references to all clusters, as expressed by that channel.

- Pick one channel's aggregate threshold image as the reference, and compute the offsets for the other 3 channels against that reference.
- Note that this requires a complete pass over all the images for the tile (4 channels x N cycles = 4N images).

Estimates, per cycle:

Thresholding is equivalent to finding the minimum and maximum neighborhood values, and comparing the pixel value to a threshold within that range.

Clocks: est 50 clocks per pixel = 190Mclocks/image

Nethra Am2045 on-chip cores: 14, (SR and SRD mix)

Nethra Am2045 RAM Unit Banks on-chip: 4 for small FIFOs

Nethra Am2045 DDR Connections: 1 (3 shared virtual connections)

DDR Bandwidth: 60MB/image

Remember that there are 4N images to process.

Computing the offsets against the reference is currently done with a very large FFT. The largest practical FFT sizes using Nethra Am2045 chips is 16K points with a single Nethra Am2045 chip, and 64K points with multiple Nethra Am2045 chips. The Nethra Am2045 MPPA architecture is best suited for FFTs no larger than 64K points. Therefore a different algorithm must be employed.

Since the results of the FFT are used to find the offset between images which represents physical displacement, a technique from motion video can be applied. The Nethra Am2045 MPPA architecture is very efficient for Sum of Absolute Difference (SAD) calculations, and can produce a 2Kx2K 16-bit SAD in approximately 8M clocks. Checking an entire offset range of +/-5 pixels in both directions would take approximately 968M clocks. As is common in video motion-estimation, it is likely that a down-scaled subset of the full image size can be used for even greater compute-density and still generate identical results.

Because this step occurs only once per tile, the performance characteristics are less important than the hardware resources consumed.

Clocks: est 1000M clocks/final threshold image

Processors: 8, mix of SR/SRD cores

RU Banks: 8 for small fifos and local storage

DDR Connections: 1 (2-4 shared, virtual connections)

DDR Bandwidth: 50MB/final threshold image

Question: What are the typical and maximum ranges of the inter-channel offsets?

Question: What other methods of offset calculation have been tried? It seems that there are likely lower-compute-cost alternatives to the FFT method.

Computing the actual offsets:

For stage positioning offsets, Cycle 1 will be used as the reference position.

- Produce a threshold image for all 4 channels, applying the inter-channel offsets, and combine them. This should contain references to all clusters, correctly positioned.
- Split this combined threshold image into sub-regions and sub-sub-regions.

- Generate the FFT signatures for each sub-sub-region.

Estimates (once per tile):

Applying offsets should be implemented entirely by adjusting the sub-region memory address, and so take virtually no time.

Thresholding is equivalent to finding the minimum and maximum neighborhood values, and comparing the pixel value to a threshold within that range.

Clocks: est 50 clocks per pixel = 190Mclocks/image

Processors: 14, SR and SRD mix

RU Banks: 4 for small FIFOs

DDR Connections: 1 (3 shared virtual connections)

DDR Bandwidth: 60MB/image

Remember that there are 4 images to process per tile.

For the sub-sub-region FFT signatures, the region size is roughly 512x512, estimated at 13Mclocks. Again, this could be off by a factor of 4, so using 50Mclocks instead to be conservative.

Estimates (once per tile):

Clocks: est 50Mclocks per sub-sub-region = 800Mclocks/image

Processors: 8, SR/SRD mix

RU Banks: 8 for small fifos and local storage

DDR Connections: 1 (2-4 shared virtual connections)

DDR Bandwidth: 50MB/image

Remember that there are 4 images to process per tile.

Per-image Computations:

For stage positioning offsets, these steps are taken:

- Produce a threshold image for all 4 channels, applying the inter-channel offsets, and combine them. This should contain references to all clusters.
- Split this combined threshold image into sub-regions and sub-sub-regions.
- Compute the offset of each sub-sub-region against the corresponding Cycle 1 threshold sub-sub-region.
- Use the median offset from the sub-sub-regions as the offset for the sub-region relative to the reference.
- Note that the first few steps are identical to those for Cycle 1, above; only the last step – the comparison – is different.

Estimates, per image:

Applying offsets should be implemented entirely by adjusting the sub-region memory address, and so take virtually no time.

Thresholding is equivalent to finding the minimum and maximum neighborhood values, and comparing the pixel value to a threshold within that range.

Clocks: est 50 clocks per pixel = 190Mclocks/image

Processors: 14, SR and SRD mix

RU Banks: 4 for small FIFOs

DDR Connections: 1 (3 shared virtual connections)

DDR Bandwidth: 60MB/image

For the sub-sub-region FFT signatures, the region size is roughly 512x512, estimated at 13Mclocks. Again, this could be off by a factor of 4, so using 50Mclocks instead to be conservative.

Clocks: est 50Mclocks per sub-sub-region = 800Mclocks/image
Processors: 8, SR/SRD mix
RU Banks: 8 for small fifos and local storage
DDR Connections: 1 (2-4 shared virtual connections)
DDR Bandwidth: 50MB/image

For image correlation, each sub-region will be offset by the associated channel offset, plus the sub-region-specific stage offset.

1.3. Object identification and intensity extraction

- Object Identification: Cluster positions are determined by finding reference contours of connected foreground pixels in the images for the first R cycles; R is tunable, and can be less than 10.

Estimates, per image:

Thresholding is equivalent to finding the minimum and maximum neighborhood values, and comparing the pixel value to a threshold within that range.

Clocks: est 50 clocks per pixel = 190Mclocks/image

Processors: 14, SR and SRD mix

RU Banks: 4 for small FIFOs

DDR Connections: 1 (3 shared virtual connections)

DDR Bandwidth: 60MB/image

Remember that there are 4 images to process per cycle, but only the first R cycles will be processed

- Once the basic contours are found, a second step is merging the contours and giving them unique identifiers. The Swift code algorithm can be implemented on MPPA fairly directly.

Estimates, per image:

Clocks: est 80 clocks per pixel = 300Mclocks/image

Processors: 10, SR and SRD mix

RU Banks: 2 for small FIFOs

DDR Connections: 1 (3 shared virtual connections)

DDR Bandwidth: 90MB/image

Remember that there are 4 images to process per cycle, but only the first R cycles will be processed

- Intensity Extraction: For each aligned image, the maximum intensity found in each cluster is recorded. This is 4 values (one per channel) for each cluster, in each cycle. In the Swift code, the clusters are stored as Run-Length encoded sets (note: for an MPPA implementation there may be more natural ways of managing the locations).

Estimates, per image:

Clocks: est 5 clocks per pixel = 19Mclocks/image

Processors: 2, SR and SRD mix

RU Banks: None

DDR Connections: 1
 DDR Bandwidth: 15MB/image

Total estimates, for all Image Processing steps, per image:

Clocks: est 5 clocks per pixel = 19Mclocks/image
 Processors: 2, SR and SRD mix
 RU Banks: None
 DDR Connections: 1
 DDR Bandwidth: 15MB/image

As noted in the beginning the Nethra Am2045 MPPA architecture appears to be best suited to the image-processing stage. The other stages are briefly described below.

Contact for Nethra Am2045 processor:

Leigh Anderson
 Director Business Development
 Nethra Inc.
 w 503-207-5715
 m 503-484-7056
 f 503-336-3743
leigha@nethra.us.com
www.nethra-imaging.us.com

Appendix 1: Detailed Estimates, per image, for single Nethra Am2045 chip

Cycles	37
Contour cycles	6

No derating factor has been applied to these numbers

Assumes pipelining + 25%

Section of document (Phase of image-processing)	1.1	1.2				1.3		
	Back. Sub.	Initial Pass	Final Thr	Thr	FFTs	Obj ID	Contours	Intensity
Clocks, Millions	75	190	0	190	800	30	48	19
Processors	7	14	8	14	8	14	10	2
RU Banks	2	4	8	4	8	4	2	0
DDR Connections	1	1	1	1	1	1	1	1
Virtual Connections	3	3	4	3	4	3	3	1
DDR Bandwidth, MB	50	60	0	60	50	9	14	15

	Am2045 Chip has:
Clocks, Millions	300

Processors	336
RU Banks	336
DDR Connections	8
DDR Bandwidth, MBps	3200

		% of Am2045 Chip Resources
Total		
Clocks, Millions	1000	333%
Processors	77	23%
RU Banks	32	10%
DDR Connections	8	100%
Virtual DDR Connections	24	n/a
DDR Bandwidth, MBps	258	8%

Seconds/image	3.33
Seconds/tile (all cycles)	493.33
Minutes/tile (all cycles)	8.22
Maximum parallelization possible	4
Minutes/tile @ 4-chip parallelization	2.1

Note: further 4-chip-set scaling could be done in a straightforward way. An available ATCA chassis can accommodate 40 chips, on 5 blades, with a PCIe backplane.

Appendix-2, Latter Two Stages

2. Base Calling

Base Calling is a separable process from Image Analysis, and may be accomplished by a completely different software config loaded into the Nethra Am2045 MPPA. Interchange between configs would be via an industry-recognized file format tbd.

The input to Base Calling is, for each cluster location on the tile at each cycle, a list of the 4 base intensities.

The output of Base Calling is the list of bases, in order, from each cluster, for the tile.

- 2.1. Crosstalk correction
- 2.2. Phasing correction
- 2.3. Chastity filtering
- 2.4. Base calling

3. Alignment

Alignment is a separable process from Base Calling, and may be accomplished by a completely different config loaded into the Nethra MPPA. Interchange between configs would be via an industry-recognized file format tbd.

The input to Alignment is the list of bases, in order, for each cluster, from all tiles on the flowcell, and a reference sequence. Note that the clusters number in the 100Ks or more. Each cluster is up to N bases long, where N is the number of cycles.

The output of Alignment is the complete DNA sequence under test, relative to the reference sequence.

References

1. Whiteford & Kelly (2008): Swift: Primary Data Analysis for Next-gen Sequencers
2. Whiteford, et al (2009): Swift: primary data analysis for the Illumina Solexa sequencing platform
3. Unknown (2007): Tech summary: Illumina's Solexa sequencing technology (found at <http://seqanswers.com/forums/showthread.php?t=21>)