프로젝트 구현 방안 보고서

**5조(장재희, 류기민, 김다훈)**

# 1. Gaussian random number generator

ziggurat algorithm은 probability density function을 세 영역(rectangular, wedge, tail)으로 나눈다. 그 다음 rejection method를 이용하여 어떤 random point가 세 영역 중에 어디에 떨어질 지 결정한다. 거의 대부분의 점들은 rectangular region에 들어가고 random variate를 생성한다. 1.5%정도 발생하는 rectangular region이 아닌 영역들에는, exponential이나 log 연산이 필요하고 반복적인 fixed point operation unit이 필요하다. rectangular region의 연산은 pipelining이 잘 되어있고 buffering scheme을 사용해 다른 영역의 연산 결과와 비교하는데 쓰인다. 참조한 논문에서는 Xilinx XC2VP30-6에서 1초당 169,000,000 개의 normally distributed random number를 생성하는데 성공하였으며, 이는 우리 프로젝트의 성능을 높이는 데에도 큰 기여를 할 것으로 생각된다.

Zhang, Guanglie, et al. "Ziggurat-based hardware Gaussian random number generator." Field Programmable Logic and Applications, 2005. International Conference on. IEEE, 2005.

# 2. Matlab Code Analysis & Simulation

## 2.1 Matlab Code Analysis

|  |
| --- |
| /////////////////////////////////////////////////////////////////////////  final\_vals = S\*exp((r-0.5\*sigma^2)\*T + sigma \* sqrt(T)\*randn(M,1));  option\_values=max(K-final\_vals,0);  present\_vals = exp(-r\*T)\*option\_values;  int = 1.96 \* std(present\_vals)/sqrt(M);  put\_value = mean(present\_vals);  ///////////////////////////////////////////////////////////////////////// |

randn(M,1)은 각 성분이 평균이 0, 표준편차가 1인 Gaussian Random Number Generator(GRNG)에 의해 generate된 값들로 이루어진 M by 1 행렬을 만든다. 그에 따라 위와 같은 식으로 final\_vals, option\_values, present\_vals 역시 M by 1 행렬이 만들어진다. 여기서 주목한 점은 final\_vals가 아무리 커도 K보다 크면 option\_values는 0이 된다. 또한 final\_vals의 값은 항상 양수임에서 실제로 present\_vals의 계산까지 data가 유효하려면 final\_vals가 0~100사이의 값을 가져야 한다는 것이다. 또한 0이 된 option\_values는 put\_value의 값에 영향을 끼치지 않을 것이고, mean과 std를 계산하는데 있어 무시해줘야 더 빠른 성능을 낼 수 있을 것이다.

## 2.2 Matlab Simulation

Term Project에 주어진 변수를 이용하여 Matlab Simulation을 해보았다. 위에서 언급한 두 가지를 중점적으로 분석한 Simulation결과를 다음과 같이 정리 할 수 있었다.

**1) 실수표현방식-다뤄야 하는 데이터의 범위**

다음과 같은 코드를 추가해 option\_values, present\_vals의 최대값을 알아보았다.

|  |
| --- |
| display(max(option\_values));  display(max(present\_vals)); |

그 결과는 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 시행1 | 시행2 | 시행3 | 시행4 | 시행5 |
| option\_values | 44.6432 | 45.5159 | 43.9242 | 45.4273 | 43.5972 |
| present\_vals | 38.4248 | 39.3759 | 37.8059 | 39.0996 | 37.5245 |

Matlab 시뮬레이션 결과 위에서 언급한 것 처럼 실제로 final\_vals의 값들은 큰 값들을 가지기도 했지만, option\_values를 계산할 때, K와 비교해 큰 값들은 모두 0으로 계산이 되었기 때문에 option\_values들의 값은 모두 0에서 50(변수 K)사이의 범위를 가졌다. option\_values에 곱해지는 값인 exp(-rT)역시 주어진 범위에서 1보다 작으므로 present\_vals의 범위 역시 0에서 50사이의 값을 가졌다. 따라서 우리가 다뤄야 하는 data의 범위가 비교적 작다는 것을 알 수 있었다. 우리가 사용할 하드웨어에는 9-bit 곱셈기 36개가 있다. 정확도를 위해서는 18-bit data를 만들어야 할 것이다. 하지만 이 때, 곱셈 한 번에 곱셈기를 4개를 사용해야 하므로 부족할 수 있을 것 같다. 하지만 시뮬레이션을 통해 우리가 다뤄야 하는 데이터의 범위가 생각보다 크지 않다는 점에서 정확도와 resource간의 trade off를 계산해 곱셈시 곱셈기를 되도록 적게 사용할 수 있도록 하는 방안을 고려해 볼 것이다.

**2) option\_values의 0들을 처리해주는 방식**

다음과 같은 코드를 추가해 option\_values에 0이 얼마나 있는지 알아보았다.

|  |
| --- |
| cnt = 0;  for j = 1:M     if option\_values(j,1)==0        cnt=cnt+1;     end  end  display(cnt); |

그 결과는 다음과 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| M=1e7 | 시행1 | 시행2 | 시행3 | 시행4 | 시행5 |
| cnt | 6367022 | 6370155 | 6366617 | 6368211 | 6369132 |

시뮬레이션 결과 총 1천만개 중 무려 636만개가 0이라는 데이터 값을 가졌다. 나중에 present\_vals의 mean과 std를 계산할 때, 0은 아무런 영향을 끼치지 않는다. 만약 0이라는 값을 특별히 다루지 않고 평균을 계산한다면 0이라는 값들을 더하는데 쓸데없는 시간을 소비할 것이고 그게 무려 63.6퍼센트나 되므로 이 경우를 잘 처리하는 것이 성능에 중요한 영향을 미칠 것이다. 이 것을 해결하기 위한 방법은 애초에 final\_vals를 계산하여 그 값이 100을 넘는다면(final\_vals의 최대값은 K, K의 최대값은 100) 이후 과정은 skip하고 다음 것을 계산하는 방식으로 설계를 하면 될 것이다.

# 3. resource 사용(SRAM)

한 번 병렬적으로 처리하는 데이터 수를 늘리거나, 아니면 데이터 수가 적어도 빠르게 처리할 수 있도록 설계하여, 둘 중에 좋은 성능을 내는 것으로 선택해야할 것이다. 이 때 한 번에 처리하는 데이터 수가 많으면 resource가 부족하게 될 것이기 때문에 SRAM을 사용할 수 있다.

# 4. 리눅스 디바이스 드라이버 작성

FPGA M3 모듈과 CPU M1 모듈이 공유하고 있는 메모리 주소(0x8800\_0000 ~ 0x9000\_0000)에서 데이터 교환이 일어나므로, 해당 메모리 주소에 필요한 계산 값을 넘겨주도록 작성한다. 필요한 상수들은 M1에서 미리 계산해서 사용한다.

# 5  JNI 라이브러리 작성

리눅스 디바이스 드라이버를 안드로이드 애플리케이션에서 사용하기 위해 지정된 프로젝트 명 아래에 필요한 함수들을 구현한다.

# 6 안드로이드 애플리케이션 작성

데이터를 입력하고 계산 결과를 출력하기 위한 안드로이드 애플리케이션을 구현한다. 시연 시에 컴퓨터와 데이터 송수신이 필요하기 때문에 해당 기능도 구현해야 한다. 프로젝트 명세서에 명시된 패키지 이름과 레이아웃에 따라 작성한다.

데이터 송수신 부분은 추가 논의가 필요하므로 여기서는 간략한 레이아웃 코드와 그 실행 결과만을 첨부하겠다.

|  |
| --- |
| <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"     xmlns:tools="http://schemas.android.com/tools"     android:layout\_width="match\_parent"     android:layout\_height="match\_parent"     android:orientation="vertical"     android:paddingBottom="@dimen/activity\_vertical\_margin"     android:paddingLeft="@dimen/activity\_horizontal\_margin"     android:paddingRight="@dimen/activity\_horizontal\_margin"     android:paddingTop="@dimen/activity\_vertical\_margin"     tools:context=".MainActivity" >  <LinearLayout     android:layout\_width="match\_parent"     android:layout\_height="0dp"     android:layout\_weight="1"     android:orientation="horizontal">     <TextView         android:id="@+id/result\_message"         android:layout\_width="0dp"         android:layout\_height="match\_parent"         android:layout\_weight="1"         android:text="@string/result\_template" />     <TextView         android:id="@+id/status\_message"         android:layout\_width="0dp"         android:layout\_height="match\_parent"         android:layout\_weight="1"         android:text="@string/status\_template" />  </LinearLayout>     <Button         android:id="@+id/connect\_toggle"         android:layout\_width="match\_parent"         android:layout\_height="wrap\_content"         android:text="@string/connect\_btn" />  </LinearLayout> |



화면 우측에는 id가 컴퓨터와의 Serial 포트 연결 상태를 보여줄 것이며, 화면 좌측에는 컴퓨터와 FPGA로의 input, output data를 보여줄 것이다. 아래의 connect 버튼을 이용하여 컴퓨터와의 serial port 연결과 연결 해제를 할 것이다.

# 7. 역할 분담

HW설계 및 작성 - 류기민, 장재희

애플리케이션 설계, 리눅스 드라이버 설계, JNI 라이브러리 설계 및 작성 -  김다훈