FPGA Acceleration of Mean Variance Framework for Optimal Asset Allocation

Ali Irturk†, Bridget Benson†, Nikolay Laptev‡, Ryan Kastner†

†Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093
{airturk, b1benson, kastner}@cs.ucsd.edu

‡Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90095
nlaptev@cs.ucla.edu

Abstract— **Asset classes respond differently to shifts in financial markets, thus an investor can minimize the risk of loss and maximize return of his portfolio by diversification of assets. Increasing the number of diversified assets in a financial portfolio significantly improves the optimal allocation of different assets giving better investment opportunities. However, a large number of assets require a significant amount of computation that only high performance computing can currently provide. Because of the highly parallel nature of Markowitz' mean variance framework (the most popular approximation approach for optimal asset allocation) an FPGA implementation of the framework can also provide the performance necessary to compute the optimal asset allocation with a large number of assets. In this work, we propose an FPGA implementation of Markowitz' mean variance framework and show it has a potential performance ratio of 221× over a software implementation.**

I. Introduction

Asset allocation is the core part of portfolio management. With asset allocation, an investor distributes his wealth across different asset classes which include different securities such as bonds, equities, investment funds, derivatives, etc. in a given market to form a portfolio. Because each asset class responds differently to shifts in financial markets, an investor can minimize the risk of loss and maximize the return of his portfolio by diversifying his assets. The goal of the portfolio manager in a financial institution is to provide the asset allocation with the greatest return for some level of risk for investors [1][2].

A portfolio manager needs to include two pieces of information to determine the best allocation for a given investor: the investor's profile and the market data. The *investor profile* includes the current asset allocation of the investor, the budget, the investment time horizon, and the investor's objectives and satisfaction indices to be able to evaluate the portfolio's performance. The *market data* include the joint distribution of the prices at the investment horizon and the implementation costs for trading these securities.

Determining the best allocation for a given investor requires solving a constrained optimization problem [3][4][5]. Convex programming problems represent a broad class of constrained optimization problems which can be solved numerically [6]; however an optimal asset allocation problem includes a large number of variables that need to be processed which requires a long computation time. Therefore, using an approximation method for the allocation optimization is crucial.

The most popular approximation approach for optimal asset allocation is *Markowitz's mean variance framework* [7]. In this framework, the investor tries to maximize the portfolio's expected return for a given risk and investment constraints. Mean variance framework is a two-step approach which approximates the solution of the optimal asset allocation problem as a tractable problem. The first step of the mean variance optimization selects efficient allocations for different risks among all the possible combinations of assets to form the efficient frontier; and the second step searches for the best allocation among all efficient allocations found in the first step.

Increasing the number of assets in a portfolio significantly improves the efficient frontier as shown in Figure 1. Adding new diversified assets to a portfolio shifts the frontier to the upper left which gives better return opportunities with less risk compared to the lower number asset portfolios. An efficient way to find an optimal allocation for small investors is to use commercially available asset allocation software: World Markets [8], Allocation Master [9], Encorr [10], PACO [11], Expert Allocator [12], Horizon [13] and Power Optimizer [14]. However financial institutions which make larger investments or control large individual investor portfolios face

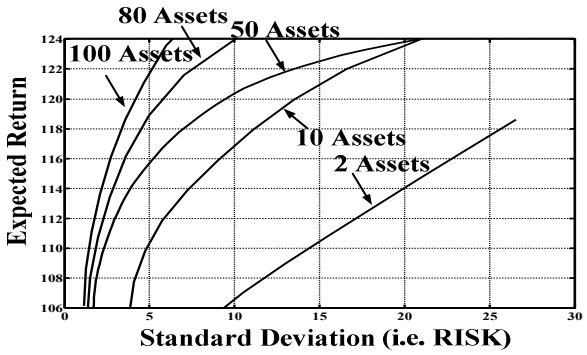

Figure 1. Increasing the number of assets in a portfolio significantly improves the efficient frontier, the efficient allocations of different assets for different risks. Adding new assets to a portfolio shifts the frontier to the upper left which gives better return opportunities with less risk compared to the lower number of assets portfolios.

more complicated problems to obtain the optimal asset allocation. Their higher number of assets and more complex diversification require significant computation that currently only high performance computing can provide.

The addition of FPGAs to the existing high performance computers can boost the application performance and design flexibility. The mean variance framework's inherent parallelism (due to many matrix computations and its use of Monte Carlo simulations) and its need for reprogramability (to allow for modifications based on different investor characteristics) make the framework an ideal candidate for an FPGA implementation. There are some previous works which consider the hardware acceleration of different financial problems, mainly concentrated on Monte-Carlo simulations, [15-21]. Zhang et al. [15] and Morris et al. [16] focused on single option pricing where Kaganov et al. [21] considered credit derivative pricing. Also interest rates and Value-at-Risk simulations are being considered by Thomas et al. in [17] and [18]. To the best of our knowledge, we are the first to propose hardware acceleration of the mean variance framework for optimal asset allocation using FPGAs.

Our major contributions are:

- A detailed description of the mean variance framework for optimal asset allocation, incorporating investor objectives and satisfaction indices used in practical implementations;
- Identification of bottlenecks for the mean variance framework which can be adapted to work in hardware;
- Design of the proposed hardware for the FPGA implementation of the mean variance framework;
- A study of potential performance improvements through simulations of the hardware architectures and a comparison between a software implementation running on two 2.4 Ghz Pentium-4 CPUs, and an FPGA architecture, showing potential performance ratios of $9.6\times$ and $221\times$ for different steps.

The rest of the paper is organized as follows: In section II, we describe the steps of the mean variance framework used for optimal asset allocation. In section III, we present our proposed implementation of the mean variance framework. Section IV presents our results in terms of timing and throughput and compares these results with a purely software implementation. We conclude and present future work in section V.

II. The Mean Variance Framework for Optimal Asset Allocation

In this section, we present the mean variance framework for optimal asset allocation. The framework is a popular two-step approach used in all practical asset allocation applications. Step 1 selects efficient allocations among all the possible combinations of assets and computes the efficient frontier. Step 2 performs a search for the best among the efficient allocations using Monte-Carlo simulations. We divide our discussion of the framework into three sections (shown in Figure 2): **A.** The computation of inputs required for Step 1 of the mean variance framework, **B.** Step 1 of the mean variance framework, and **C.** Step 2 of the mean variance framework. Note that all equations listed in the following subsections are found in [1] unless otherwise specified.

A. Computation of the Required Inputs

The expected values of the market vector $E\{M\}$ and the respective covariance matrix $Cov\{M\}$ are needed as inputs to the mean variance framework. Calculating these inputs requires the use of already known publically available data: prices, standard deviation, and covariances plus the investor's objective, number of securities, N_s, reference allocation and horizon τ (as shown in Figure 2 (a)). Calculating $E\{M\}$ and $Cov\{M\}$ using these investor and market parameters requires the 5 stage procedure (shown in Figure 3) explained in detail below [1].

1. Detection of the invariants, $X_{t,\tilde{\tau}}$

Invariants are identical repetitions in the market within a given estimation interval, $\tilde{\tau}$. The estimation interval, $\tilde{\tau}$, is different than the horizon τ, which was mentioned before. The estimation interval, $\tilde{\tau}$, refers to the time which we suspect a repetition in data, where investment horizon, τ, refers to the time the investor plans to invest. Detection of the invariants is an essential step and linear return of stocks $L_{t,\tilde{\tau}} = \frac{P_t}{P_{t-\tau}} - 1$, or compounded return of stocks $C_{t,\tau} = \ln\left(\frac{P_t}{P_{t-\tau}}\right)$ can be used as invariants for the market. We chose to use compounded return of stocks.

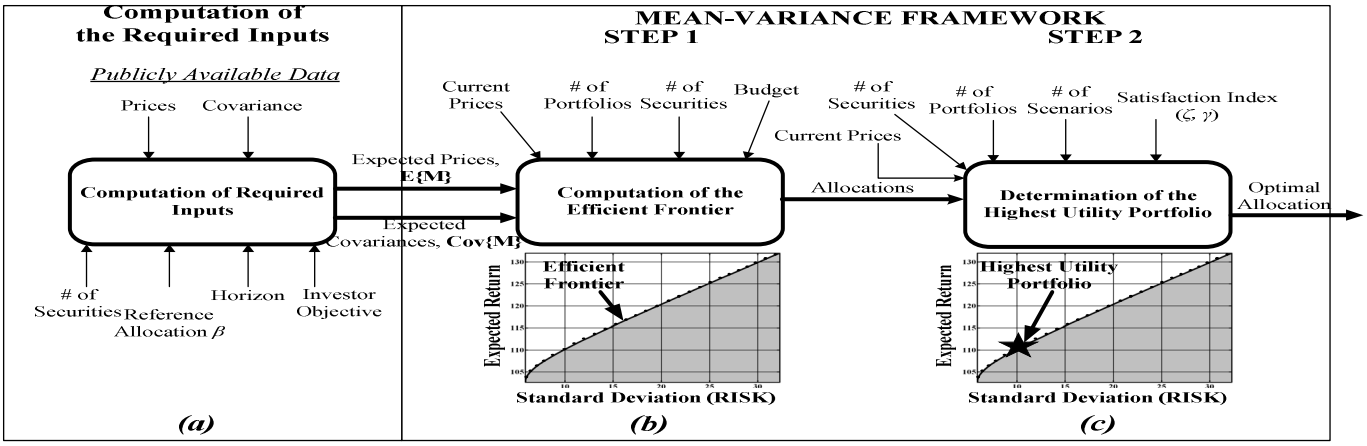

Figure 2. The required steps for optimal asset allocation are shown in (a), (b) and (c). After the required inputs to the mean variance are generated in (a), computation of the efficient frontier and determination of the highest utility portfolio are shown in (b) and (c) respectively. This figure also presents the inputs and outputs provided to the user.

2. *Determination of the distribution of the invariants*

We can determine the distribution of the invariants based on estimators (maximum likelihood estimators, nonparametric estimators etc.) based on current market information. As an example, we assume that the invariants $X_{t,\tilde{\tau}} = C_{t,\tilde{\tau}}$ are multivariate normal distribution with $N(\hat{\mu}, \hat{\Sigma})$ where $\hat{\mu}$ and $\hat{\Sigma}$ are vectors of sample mean and covariance matrix respectively.

3. *Projection of the invariants $X_{t,\tilde{\tau}}$ to the investment horizon to obtain the distribution of $X_{T+\tau,\tau}$*

After the determination of the distribution of the invariants $X_{t,\tilde{\tau}}$ in an estimation interval, $\tilde{\tau}$, we project them to the investment horizon $X_{T+\tau,\tau} \sim N(\frac{\tau}{\tilde{\tau}}\hat{\mu}, \frac{\tau}{\tilde{\tau}}\hat{\Sigma})$. Furthermore we use this distribution to determine the distribution of the market prices, $P_{T+\tau}$.

4. *Computation of the expected return $E\{P_{T+\tau}\}$, and the covariance matrix $Cov\{P_{T+\tau}\}$ from the distribution of the market prices.*

We use the characteristic function of the compounded returns to formulize the expected returns as

$$E\left\{P_{T+\tau}^{(n)}\right\} = P_T^{(n)} e^{\frac{\tau}{\tilde{\tau}}\left(\hat{\mu}_n + \frac{\hat{\Sigma}_{nn}}{2}\right)} \quad (1)$$

and covariance matrix of the market as:

$$Cov\left\{P_{T+\tau}^{(m)}, P_{T+\tau}^{(n)}\right\} =$$

$$P_T^{(m)} P_T^{(n)} e^{\frac{\tau}{\tilde{\tau}}(\hat{\mu}_m + \hat{\mu}_n)} e^{\frac{1\tau}{2\tilde{\tau}}(\hat{\Sigma}_{mm} + \hat{\Sigma}_{nn})} \left(e^{\frac{\tau}{\tilde{\tau}}(\hat{\Sigma}_{mn})} - 1\right) \quad (2)$$

5. *Computation of the expected return $E\{M\}$, and the covariance matrix $Cov\{M\}$ of the market vector*

An investor objective is a function for which every investor desires the largest value as an output of that function. There are different objectives such as absolute wealth, relative wealth and net profits [1]. An *absolute wealth* investor tries to maximize the value of the portfolio in the investment horizon. A *relative wealth* investor tries to achieve better portfolio return compared to a reference portfolio where the reference portfolio is denoted as β with γ as a normalization factor. A *net profits* investor always tries to increase the value of the portfolio compared to the value of the portfolio today. The specific forms of the equations for these objectives are shown in Table I (a).

These different objectives can be seen as a linear function of the investor's allocation α, and the market vector M, shown as follows :

$$\psi_\alpha = \alpha M \quad (3)$$

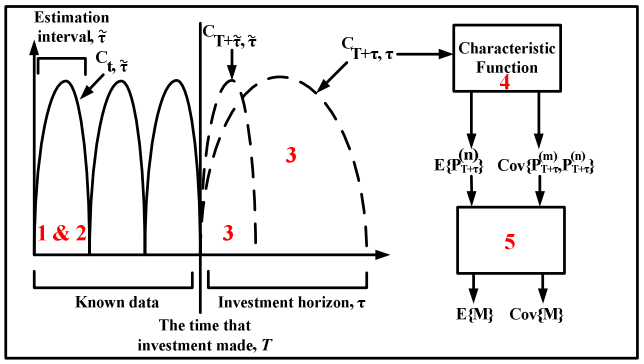

Figure 3. The procedure to generate required inputs is described. The numbers 1-5 refers to these computation steps which are explained in subsections in more detail.

M is a transformation of the market prices at the investment horizon as:

$$M \equiv \mathrm{a} + BP_{T+\tau} \quad (4)$$

where a and B are a suitable conformable vector and an invertible matrix respectively. These generalized forms of investor objectives are also shown in Table I (b) with different a and B values where $\gamma(\alpha) \equiv \frac{w_T(\alpha)}{w_T(\beta)}$ (Normalization factor), $K \equiv I_N - \frac{p_T \beta'}{\beta' p_T}$ and I_N is identity matrix . Computation of the market vector combines the expected returns and covariance matrix with the investor objectives using different a and B values for different investor objectives which is shown as :

$$E\{M\} = \mathrm{a} + \mathrm{B}E\{P_{T+\tau}\} \quad (5)$$

$$Cov\{M\} = \mathrm{B}Cov\{P_{T+\tau}\}\mathrm{B}' \quad (6)$$

Notice that each step requires the financial analyst to make assumptions (such as what type of invariant distribution to assume, and what estimation interval to use). Each assumption affects the outcome of the computation and hence each of the five steps described is a broad research area in economics. For our purposes we use the following assumptions with the knowledge these could be easily changed: we use the past 3 years of the data with 1 week estimation interval. We use compounded returns of stocks as market invariants and assume that they are multivariate random variables. We assume our estimation horizon is 1 year.

B. Mean Variance Framework Step 1: Computation of the Efficient Frontier

Computing the efficient frontier, the efficient allocations of different assets for different risks, is the first step of the mean variance framework (Figure 2(b)). The inputs to this step are current prices (already known), expected prices, *E{M},* and

TABLE I
DIFFERENT INVESTOR OBJECTIVES, SPECIFIC AND GENERALIZED FORMS

	Standard Investor Objectives		
	Absolute Wealth	**Relative Wealth**	**Net Profits**
(a) Specific Form	$\psi_\alpha = W_{T+\tau}(\alpha)$	$\psi_\alpha = W_{T+\tau}(\alpha) - \gamma(\alpha)W_{T+\tau}(\beta)$	$\psi_\alpha = W_{T+\tau}(\alpha) - w_T(\alpha)$
(b) Generalized Form	$\mathrm{a} \equiv 0, \quad B \equiv I_N$ $\boldsymbol{\psi_\alpha = \alpha' P_{T+\tau}}$	$\mathrm{a} \equiv 0, \quad B \equiv K$ $\boldsymbol{\psi_\alpha = \alpha' K P_{T+\tau}}$	$\mathrm{a} \equiv -p_T, \quad B \equiv I_N$ $\boldsymbol{\psi_\alpha = \alpha'(P_{T+\tau} - p_T)}$

expected covariance matrix, *Cov{M},*(which are calculated as described in section II-A), number of portfolios, N_p, number of securities, N_s and investor's budget. This step calculates N_p amount of efficient portfolios. These different portfolios create the curve in Figure 2 (b) which is called the efficient frontier.

Assume an investor who purchases α_n units of the n-th security in a market of N securities at time T (the time that the investment is made). If $P_T^{(n)}$ and α denote the price of the n-th security at the time T and the allocation at the time the decision is made respectively, the value of the portfolio is calculated as :

$$W_T(\alpha) \equiv \alpha P_T \tag{7}$$

However, the market prices of the securities are multivariate random variables at the investment horizon, therefore the portfolio is a random variable which can be seen as :

$$W_{T+\tau}(\alpha) \equiv \alpha' P_{T+\tau} \tag{8}$$

where α' refers to the allocation at the horizon. Because the portfolio's value is a random value since the market prices are unknown, the expected prices at the investment horizon $E\{P_{T+\tau}\}$ and the covariance matrix $Cov\{P_{T+\tau}\}$ need to be computed and then investor objective function needs to be included to give us $E\{M\}$ and $Cov\{M\}$ (These calculations are shown in the previous section). The efficient frontier is then found by maximizing the investor objective value by a constrained variance. This computation can be seen as :

$$\alpha(v) \equiv arg \max_{\substack{\alpha \in \mathbb{C} \\ \alpha' Cov\{M\}\alpha = v}} \alpha' E\{M\} \quad , v \geq 0 \tag{9}$$

where an investor's objective value and variance is calculated as follows:

$$E\{\psi_\alpha\} = \alpha' E\{M\} \tag{10}$$

$$Var\{\psi_\alpha\} = \alpha' Cov\{M\}\alpha \tag{11}$$

With the efficient frontier depending on how much risk an investor wants to face, there is a corresponding expected return. The region which is below the black curve (the shaded region in Figure 2(b)) corresponds to the achievable risk-return space for the specific frontier which includes at least one portfolio constructible from the investments that has the risk and return corresponding to that point. The upper region is the unachievable risk-return space. The black curve running along the top of the achievable region is the efficient frontier. The portfolios that correspond to points on that curve are optimal according to equation (9).

C. Mean Variance Framework Step 2: Computing the Optimal Allocation

Now that we have generated the inputs for the mean variance framework and used these inputs to compute the efficient frontier, we have to consider satisfaction indices to determine which 'point' along the efficient frontier represents the optimal allocation for the given investor. The required inputs to this step are the allocations computed in step 1, current prices, number of portfolios, N_p, number of securities, N_s, number of scenarios, N_m, and investor satisfaction index (as shown in Figure 2 (c)).

The investor objective function produces one value. However this value is random since the market prices at the investment horizon are stochastic and therefore the market vector, M, contains random variables. Therefore, using the investor function alone does not allow us to select the optimal allocation because we have no way of determining which random value output is 'better' for the investor than another. Therefore we need to compute the expected value of the investor objective value by introducing satisfaction indices [1]. Satisfaction indices represent all the features of a given allocation with one single number and quantify the investor's satisfaction. Therefore, an investor prefers an allocation to the other if it provides more satisfaction. There are mainly three different classes of indices being used to model the investor's satisfaction: certainty-equivalent, quantile and coherent indices. We use certainty-equivalent indices because they are based on a concave function and promote diversification [1].

Certainty-equivalent indices are represented by the investor's utility function and objective. A *utility function* $u(\psi)$ is defined for an investor to explain his enjoyment. There are different utility functions which we can use to represent an investor's satisfaction such as exponential, quadratic etc. Even though this function is specific for every investor, it is possible to investigate the most commonly used functions and generalize them [1]. We show these different utility functions in Table II. To generalize the creation of utility functions, we use Hyperbolic Absolute Risk Aversion (HARA) class of utility functions which are specific forms of the Arrow-Pratt risk aversion model and defined in [1, 22] as

$$A(\psi) \equiv \frac{\psi}{\gamma\psi^2 + \zeta\psi + \eta} \tag{12}$$

where $\eta \equiv 0$. The HARA class of utility functions gives us most of the utility functions by varying the constants, ζ and γ as shown in Table II.

Therefore, an investor compares different allocations using the index of satisfaction and chooses the maximum value as the optimal asset allocation. Computing the optimal allocation is a maximization problem using different market scenarios since market values are uncertain and its analytical solution is not possible in many practical implementations [1]. Therefore approximation methods are employed for finding the best allocation on the efficient frontier. To solve this problem with approximations, a large number of market scenarios are simulated through Monte-Carlo simulations.

TABLE II
DIFFERENT UTILITY FUNCTIONS FOR SATISFACTION INDICES

Utility Functions				
Exponential Utility ($\zeta > 0$ *and* $\gamma \equiv 0$)	**Quadratic Utility** ($\zeta > 0$ *and* $\gamma \equiv -1$)	**Power Utility** ($\zeta \equiv 0$ *and* $\gamma \geq 1$)	**Logarithmic Utility** ($lim_{\gamma \to 1}\gamma$)	**Linear Utility** ($lim_{\gamma \to \infty}\gamma$)
$\boldsymbol{u(\psi) = -e^{-\frac{1}{\zeta}\psi}}$	$\boldsymbol{u(\psi) = \psi - \frac{1}{2\zeta}\psi^2}$	$\boldsymbol{u(\psi) = \psi^{1-\frac{1}{\gamma}}}$	$\boldsymbol{u(\psi) = ln\psi}$	$\boldsymbol{u(\psi) = \psi}$

III. Implementation of the Mean Variance Framework

Now that we have described how optimal asset allocation works, we now discuss our proposed implementation of the mean variance framework. We first present a series of figures to provide the motivation for our implementation and determine the bottlenecks of optimal asset allocation. We then describe the proposed architectures and possible ways to benefit from their inherent parallelism.

A. Implementation Motivation

As previously shown in Figure 1, increasing the number of securities in a portfolio allows the investor to achieve better investment opportunities, thus our goal is to allow for a large number of diversified securities in a portfolio. But how much computation time does increasing the number of securities add to the computation of the optimal asset allocation? To address this question we looked at how varying the number of securities affected the computation time in relation to number of portfolios and number of scenarios, two other important parameters that affect computation time (an increase in the number of portfolios increases the number of points on the efficient frontier and increasing the number of scenarios increases the number of runs of the Monte-Carlo simulation). Figure 4 (a) and (b) compare number of securities, N_s, versus number of portfolios, N_p, and number of portfolios, N_p, versus number of scenarios, N_m, respectively. By looking at the slopes of the lines in the figures it can be easily seen that N_s dominates computation time (has a steeper slope) over N_p (a), and N_p dominates computation time over N_m (b). These results suggest that the number of securities is the most computationally time sensitive input to the optimal asset allocation problem, thus if a large number of securities are to be allowed as input to the framework, a faster implementation must be developed.

To identify the bottlenecks of the computation of the optimal asset allocation, we look at the runtime of each solution step (**1.** generation of the required inputs, **2.** Step 1 of the mean variance framework, and **3.** Step 2 of the mean variance framework) with respect to varying the number of securities (Figure 5 (a)) and number of portfolios (Figure 5 (b)). As can be seen from Figure 5 (a), the generation of the required inputs does not consume a significant amount of time, thus it is best to keep this implementation step in software if the computation cannot be parallelized. On the other hand, step 1 and 2 of the mean-variance framework consume a significant amount of time providing the motivation for an alternative implementation. It is also important to note there is a cutoff point between step 1 and 2, showing that the computational time for step 1 becomes more significant after 60 securities. In (b), we only compare step 1 and 2 for different number of portfolios (because we already determine that the computational time for the generation of required steps is not significant), and we conclude that most time consuming part is step 1.

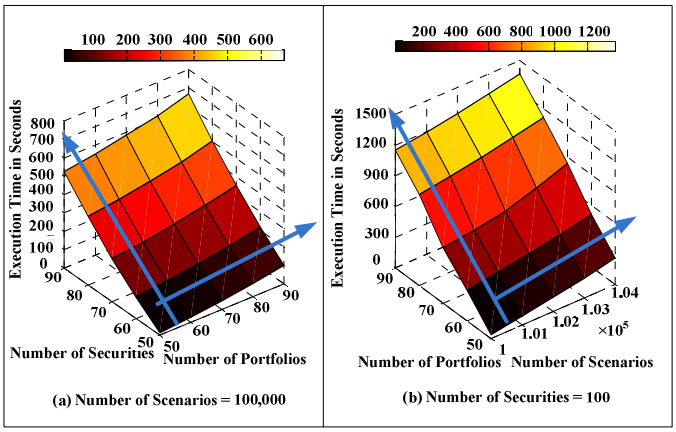

Figure 4. To determine the computation time of different variables, we compare number of securities, N_s, versus number of portfolios, N_p, and number of portfolios, N_p, versus number of scenarios, N_m, respectively. By looking at the slopes of the lines in the figures it can be easily seen that N_s dominates computation time (has a steeper slope) over N_p (a), N_p dominates computation time over N_m.

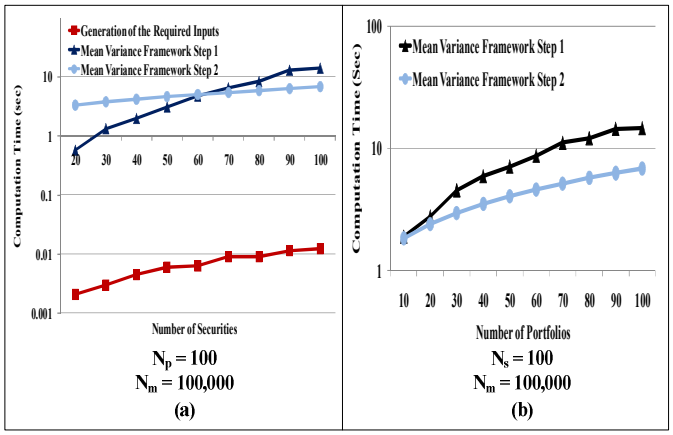

Figure 5. Identification of the bottlenecks in the computation of the optimal asset allocation. We run two different test while holding all but one variable constant. We determined that generation of the required input does not consume significant amount of time. On the other hand, step 1 and 2 of the mean variance framework consumes significant amount of time.

B. Hardware/Software Interface

As determined in section III-A, Step 1 and Step 2 of the mean variance framework are the bottlenecks for computing the optimal asset allocation. FPGA implementations can provide a substantial performance improvement for processes that can be easily parallelized. Fortunately, finding the maximum return for different risk values to create the efficient frontier (Step 1) and implementing Monte-Carlo simulations to apply different market scenarios (Step 2) can be easily parallelized making them good candidates for hardware implementations. Although the generation of required inputs is not a bottleneck for optimal asset allocation, further performance improvement can be gained by implementing phase 5 of this step (computation of $E\{M\}$ and $Cov\{M\}$) which includes parallizable matrix computations in hardware. Thus, our implementation combines software (a Host PC) to compute phases 1-4 of the generation of required inputs and hardware (FPGA) to compute phase 5 of the generation of required inputs and step 1 and step 2 of the mean variance framework to obtain maximal performance gain.

Because our implementation combines hardware and software, we must pay particular attention to the hardware/software interface, especially to the data that needs to be transferred, to insure we do not lose the performance gain we added through the hardware software separation. The

information that needs to be transferred between the software and hardware are current prices, expected prices, and expected covariances which are of the dimensions $N_s \times 1$, $N_s \times 1$ and $N_s \times N_s$ respectively. In the following subsections, we present our architectural design and parallelization possibilities for the generation of the required inputs phase 5, mean variance framework step 1 and 2.

C. Generation of Required Inputs – Phase 5

We implement "*Market Vectors Calculator IP Core*" for the calculation of phase 5 in the generation of the required inputs (Figure 6 (a)). This IP Core can compute three different objectives: absolute wealth, relative wealth and net profits or any combination of these which are described in subsection II-A. This IP Core includes the *K building block* which computes the constant matrix *K* and used if the investor's objective is relative wealth. The required inputs to this hardware are current prices, P_T, reference allocation, β', identity matrix, I_N, and expected returns, $P_{T+\tau}$,. We use two control inputs: *cntrl_a* and *cntrl_b* to select the desired investor profile. These control relationships are described as:

	Control Inputs	
Investor Objective	*cntrl_a*	*cntrl_b*
Absolute Wealth	0	0
Relative Wealth	1	0
Net Profits	0	1

After these control units are given, *E{M},* the market vectors at the investment horizon, is calculated. Figure 6 (b) shows how the *Market Vectors Calculator IP Core* can be easily parallelized. *Cov{M}*, computed by equation (6) is only needed when the investor objective is relative wealth. Because it also includes many matrix multiplications and accumulations, a similar parallelized hardware can be implemented.

D. Hardware Architecture for Mean Variance Framework Step 1

Mean variance framework step 1 is a constrained maximization problem which is shown in equation (9). This step receives market vectors, $E\{M\}$ and $Cov\{M\}$ as inputs and maximizes the expected return for a specific standard

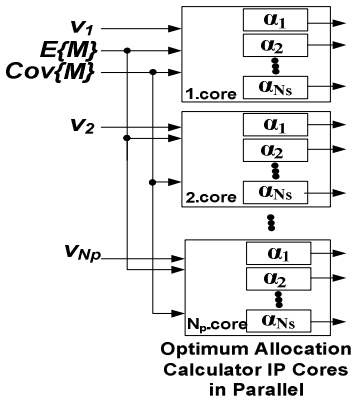

Figure 7. Parallel optimum allocation calculator IP Cores.

deviation (risk) to find the efficient portfolio. This maximization problem needs to be solved for different risks number of portfolios, N_p, times. A simple example of this maximization problem can be seen as:

$$\alpha(v) \equiv arg \max_{\alpha' Cov\{M\}\alpha = v} \alpha' E\{M\} \quad , v \geq 0 \qquad (13)$$

where two possibly important constraints: the budget of the investor and $\sum_{i=1}^{N_s} a_i = 1$ are not added the for ease of understanding.

A popular approach to solve constrained maximization problems is to use the *Lagrangian multiplier method* [23] which introduces an additional variable, λ, to equalize the number of equations and number of unknowns. The equations for the solution of the equation (13) for 2 securities can be seen as:

$$\mathcal{L} = a' E\{M\} + \lambda(v - \alpha' Cov\{M\}\alpha) \qquad (14)$$

$$\mathcal{L} = [a_1\ a_2]\begin{bmatrix} P_1 \\ P_2 \end{bmatrix} + \lambda(v - [a_1\ a_2]\begin{bmatrix} Cov_{11} & Cov_{12} \\ Cov_{21} & Cov_{22} \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}) \qquad (15)$$

$$\frac{\partial \mathcal{L}}{\partial a_1} = P_1 - \lambda[2a_1 Cov_{11} + a_2(Cov_{21} + Cov_{12})] = 0 \qquad (16)$$

$$\frac{\partial \mathcal{L}}{\partial a_2} = P_2 - \lambda[2a_2 Cov_{22} + a_1(Cov_{21} + Cov_{12})] = 0 \qquad (17)$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = a_1^2 Cov_{11} + a_1 a_2(Cov_{21} + Cov_{12}) + a_2^2 Cov_{22} = v \qquad (18)$$

By solving three equations for three unknowns, a_1, a_2, λ,

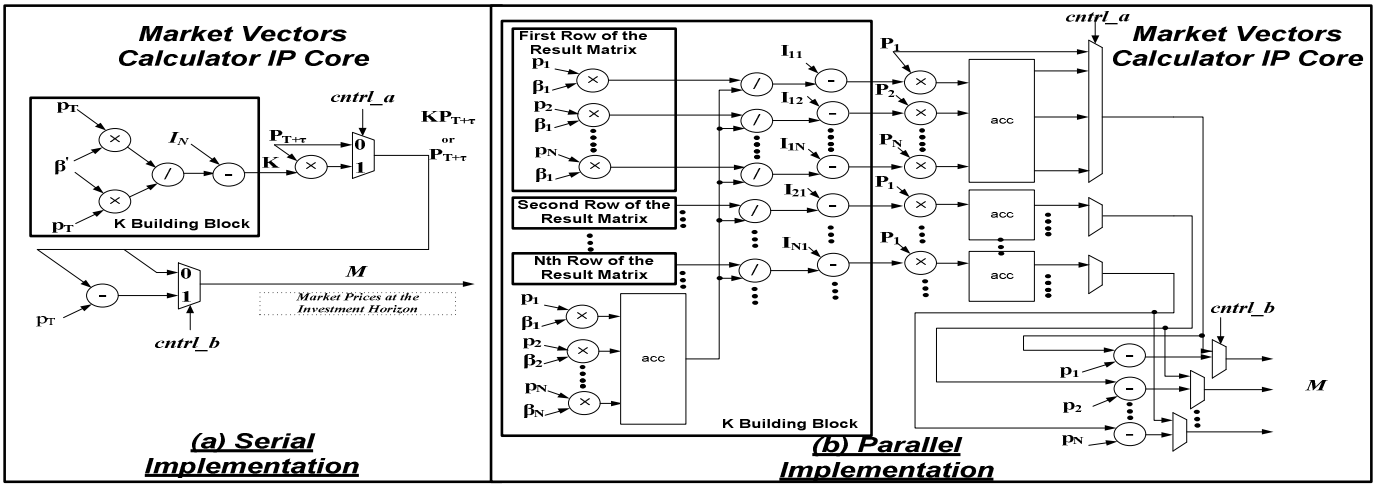

Figure 6. Parameterizable hardware architecture for the generation of the required inputs – phase 5. Two different architectures are presented as serial and fully parallel. As can be seen from the parallel architecture, phase 5 has very high potential for the parallel implementation, therefore a good candidate for decreasing the computational time of the optimal asset allocation.

one can derive the optimal α_1 and α_2 values where calculation of these values can be written as functions of the known constants such as $v\,(Risk)$, $P_1$and Cov_{22}. A number of securities, N_s, amount of functions need to be computed for determination of the efficient allocation for a given risk. These equations will be the same for different risks and hence can be easily parallelized (as shown in Figure 7). There are different α calculator blocks in every core. This core can be used serially by applying different variances as inputs or can be parallelized since the equations these cores include are the same.

E. Hardware Architecture for Mean Variance Framework Step 2

After computing the efficient frontier, we determine the highest utility allocation (optimal allocation) among these different allocations using satisfaction indices. Computing the optimal allocation is a maximization problem by simulating a large amount of market scenarios through Monte-Carlo simulations. The *Satisfaction Function Calculator IP core* (Figure 8 (a)) has required inputs of the investor objective function values ψ, and the constants ζ and γ which are defined in equation (12). The *Satisfaction Function Calculator IP core* can evaluate linear, logarithmic, exponential, quadratic, and power utility functions. The control input, *cntrl_c*, defines which utility to use.

For the determination of the highest utility allocation, the *Monte-Carlo block* and the *Utility Calculation Block* (as part of the Satisfaction Function Calculator block) are run number of simulations, N_m, times. The whole *Satisfaction Function Calculator IP core* is then run number of portfolios, N_p, times. Therefore, the *Monte-Carlo block*, *Utility Calculation Block*, and *Satisfaction Function Calculator IP core* can be easily parallelized a maximum of N_m, N_m and N_p times respectively as shown in Figure 8 (b).

IV. RESULTS

In this section, we investigate potential speed-ups for the mean variance framework using simulations of the hardware architectures we described in Section III. We concentrate on "Generation of the required inputs – Phase 5" and "the mean variance framework – step 2." We consider serial and different level of parallel implementations of these steps and compare our results with a software implementation running on two 2.4 Ghz Pentium-4 CPUs (every test is run 1000 times and average runtime is presented). We use 32 bit fixed-point arithmetic for our implementations and assume that our clock frequency achieves 200 MHz. The complexity of the mean variance framework step 1 increases dramatically with increased securities, and hence its potential runtime cannot be determined until we investigate alternative parallelism methods (such as employing Monte-Carlo simulations) and hence is not presented.

As can be seen from Figure 6, the market vector calculator IP Core can be implemented with different levels of parallelism levels where we are bound by hardware resources rather than by the parallelism that this step offers. The *serial implementation* of this step (no parallelism exploited) performs poorly compared to the software implementation. The *parallel implementation* uses a reasonable parallelism level by employing N_s number of arithmetic resources in parallel: for 50 securities there are 50 multipliers, dividers, subtractors etc. This level of parallelism achieves a potential performance ratio between $6\times$ and $9.6\times$ compared to the software implementation. A *fully parallel implementation* which might not be realistic due to hardware limitations, presents a best potential bound offering a performance ratio $629\times$ (for 50 securities). This comparison is shown in Figure 9 (a).

We investigate the difference in timing for mean variance framework step 2 in Figure 9(b). We use 100,000 scenarios, N_m, for Monte-Carlo simulations and 50 portfolios, N_p, to evaluate. We present two parallel architectures, parallel 1 employs 10 Satisfaction Function Calculator blocks where each consists of 1 Monte-Carlo block with 10 multipliers and 10 Utility Function Calculator blocks. Parallel 2 employs 10 Satisfaction Function Calculator blocks where each consists of 1 Monte-Carlo block with 20 multipliers and 20 Utility Function Calculator blocks. Parallel 1 and Parallel 2 offer a potential performance ratio between $151\times$ and $221\times$ and between $302\times$ and $442\times$.

As can be seen from the potential performance ratios, both "Generation of the required inputs – phase 5" and "mean variance framework – step 2" offer significant speed-up when parallelized and implemented in hardware.

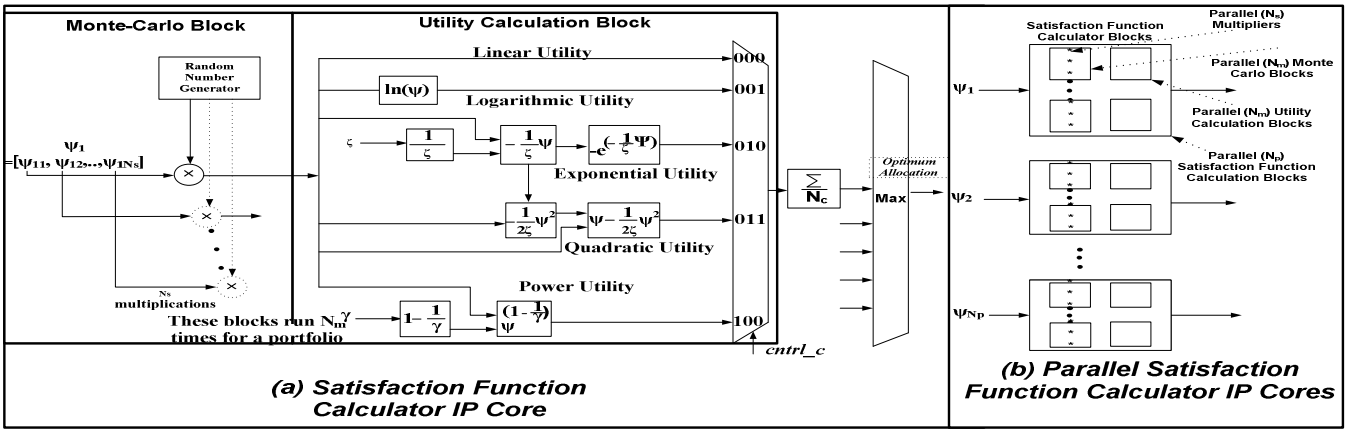

Figure 8. Parallel parameterizable hardware architecture for the mean variance framework step 2. The *Monte-Carlo block*, *Utility Calculation Block*, and *Satisfaction Function Calculator IP core* can be easily parallelized a maximum of N_m, N_m and N_p times respectively.

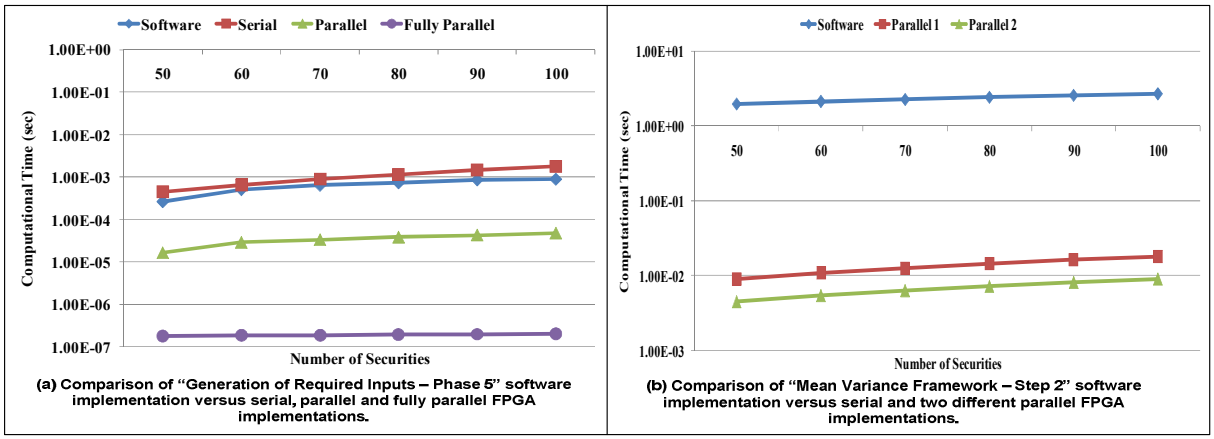

Figure 9. Possible speed-ups for "generation of the required inputs - phase 5" and mean variance framework Step 2.

V. Conclusions and Future Work

The addition of FPGAs to the existing high performance computers can boost an application's performance and design flexibility. The mean variance framework's inherent parallelism in its solution steps (due to many matrix computations and its use of Monte Carlo simulations) and its need for reprogramability (to allow for modifications based on different investor characteristics) make the framework an ideal candidate for an FPGA implementation. In this work, we are the first to propose hardware acceleration of optimal asset allocation through an FPGA implementation of Markowitz' mean variance framework. We concentrate on "Generation of the required inputs – Phase 5" and "Mean-variance Framework – Step 2" in this work and present a study of potential performance improvements through simulations of the hardware architectures. We provide a comparison between a software implementation running on two Pentium-4 CPUs, and an FPGA architecture, showing potential performance gains of 9.6 × for Phase 5 and 221 × for Step 2.

We are currently working on a parameterizable hardware implementation of these steps with fixed-point and floating point arithmetic. There are different architectural choices (parallelism levels) to implement these steps and we are bound with the hardware resources instead of parallelism offered. Thus we follow a design methodology which we can automatically generate and optimize different architectures and perform a design space exploration to find efficient implementations by generating libraries and working on a tool [24-27] which can perform these tasks.

References

[1] A. Meucci, "Risk and Asset Allocation," *Springer Finance Press*, 2005.

[2] F. Fabozzi, "Handbook of Portfolio Management," *Frank J. Fabozzi Associates*, New Hope, Pennsylvania, 1998.

[3] M. Lobo, L. Vandenberghe, S. Boyd and H. Lebret, " Applications of second order cone programming," Linear Algebra and its Applications, *Special Issue on Linear Algebra in Control, Signals and Image Processing* 284, 193-228, 1998.

[4] A. Ben-Tal and A. Nemirovski, "Optimal Design of Engineering Structures," *Optima* pp. 4-9.

[5] S. Boyd and L. Vandenberghe, "Convex Optimization", *Cambridge University Press*, 2004.

[6] Y. Nesterov, and A. Nemitovski, "Interor-Point Polynomial Algorithms in Convex Programming", *Society for Industrial and Applied Mathemetics*, 1995.

[7] H. M. Markowitz, "Portfolio Selection: Efficient Diversification of Investments", 2nd Edition, 1991.

[8] http://www.barra.com

[9] http://www.sungard.com/AllocationMaster/

[10] http://www.ibbotson.com

[11] http://www.northinfo.com

[12] http://invest-tech.com/allocator.html

[13] http://www.wilshire.com

[14] http://wilsonintl.com

[15] G.L. Zhang, P.H.W. Leong, C.H. Ho, K.H. Tsoi, C.C.C. Cheung, D.-U. Lee, R.C.C. Cheung, W. Luk, "Reconfigurable acceleration for Monte Carlo based financial simulation," *2005 IEEE International Conference on Field-Programmable Technology, 2005.* vol., no., pp. 215-222, 11-14 Dec. 2005.

[16] G.W. Morris, M. Aubury, "Design Space Exploration of the European Option Benchmark using Hyperstreams," *International Conference on Field Programmable Logic and Applications, 2007.* vol., no., pp.5-10, 27-29 Aug. 2007.

[17] D.B. Thomas, J.A. Bower, W. Luk, "Automatic Generation and Optimisation of Reconfigurable Financial Monte-Carlo Simulations," *IEEE International Conf. on Application-specific Systems, Architectures and Processors, 2007.*, vol., no., pp.168-173, 9-11 July 2007.

[18] D.B. Thomas, W. Luk, "Sampling from the Multivariate Gaussian Distribution using Reconfigurable Hardware," *15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2007.* vol., no., pp.3-12, 23-25 April 2007.

[19] D.B. Thomas, W. Luk, "Credit Risk Modelling using Hardware Accelerated Monte-Carlo Simulation," In Proceedings of *Field-Programmable Custom Computing Machines 2008*.

[20] N. A. Woods, and T. VanCourt, "FPGA Acceleration of Quasi-Monte Carlo in Finance," *accepted for publication in Proceedings of Field Programmable Logic and Applications (FPL) 2008*.

[21] A. Kaganov, A. Lakhany and P. Chow, "FPGA Acceleration of Monte-Carlo Based Credit Derivative Pricing," *Accepted for publication in Proceedings of Field Programmable Logic and Applications (FPL) 2008*.

[22] M. LiCalzi and A. Sorato, "The Pearson system of utility functions," *Game Theory and Information 0311002*, EconWPA 2003.

[23] W. Nicholson, "Microeconomic Theory: Basic Principles and Extensions," South Western Educational Publishing, 2004.

[24] A. Irturk, B. Benson, and R. Kastner, **"**An Optimization Methodology for Matrix Computation Architectures," *submitted for review, Design, Automation & Test in Europe (DATE) 2009*.

[25] A. Irturk, B. Benson, S. Mirzaei, and R. Kastner, **"**GUSTO: An Automatic Generation and Optimization Tool for Matrix Inversion Architectures," *submitted for review, Transactions on Embedded Computing Systems.*

[26] A. Irturk, B. Benson, and R. Kastner, "Automatic Generation of Decomposition based Matrix Inversion Architectures," *accepted for publication in Proceedings of International Conference on Field-Programmable Technology (ICFPT) 2008*.

[27] A. Irturk, B. Benson, S. Mirzaei and R. Kastner, "An FPGA Design Space Exploration Tool for Matrix Inversion Architectures," *accepted for publication in Proceedings of IEEE Symposium on Application Specific Processors (SASP) 2008*.