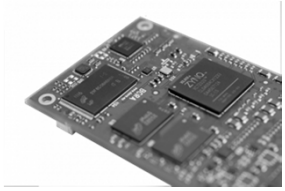
MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

MPSoC'2014
July 7-11, 2014
Margaux, France

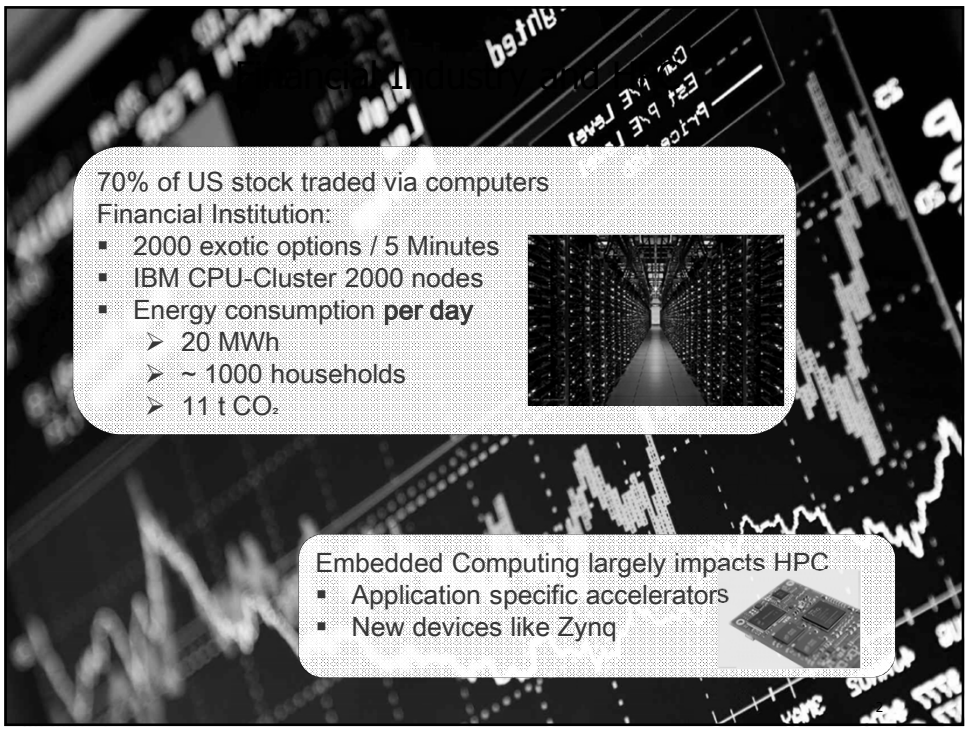
A Runtime Reconfigurable Architecture for Monte Carlo Option Pricing in the Heston Model

HPC on embedded computing devices - a Case Study -



TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

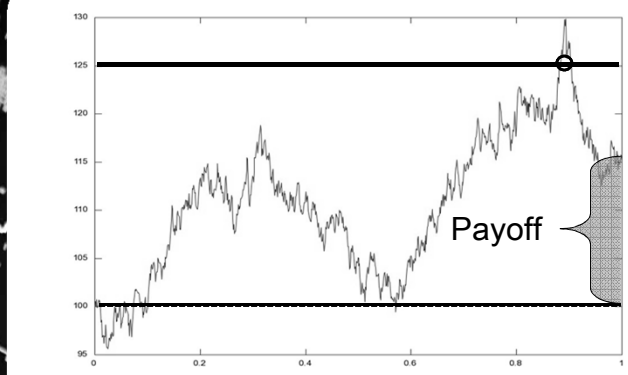
Norbert Wehn
wehn@eit.uni-kl.de



Pricing of European Call Options with Barrier

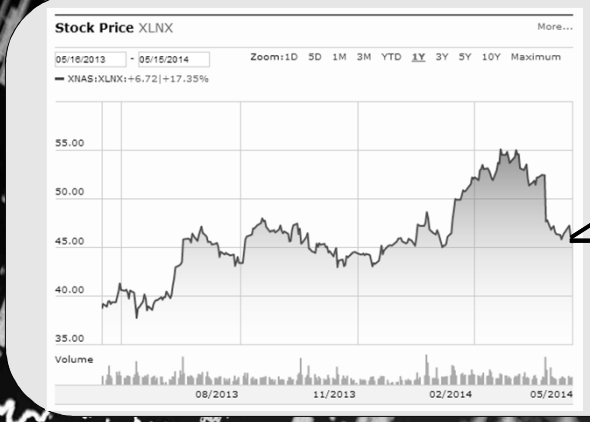
Fast calculations of market prices for all sort of financial products e.g. options
Option: financial contract between buyer and seller

- Buyer: option to buy a share from a seller at a certain time for a certain price
- Finance market: no arbitrage, i.e. making profit with no risk not possible
- Option is not for free ⇒ option price = discounted expected payoff



Source: publik16 / www.flickr.com

Pricing of European Call Options for Market Models



7th July 2014

Price: 49.25
Profit: 4.02

$45.23
Right to buy

Black-Scholes Model: $dS_t = \mu S_t d_t + V_t dW_t$

- S_t: price of the asset/share
- μ: drift, V_t: volatility, W_t: Brownian motion

Heston market model (1993)

- Volatility in BS model is constant over time
- Crash situations can not be modeled
- State-of-the-art: Heston model
- BUT: in general no closed analytical solution

Monte Carlo simulations mandatory

$$dS_t = \mu S_t d_t + \sqrt{V_t} S_t dW_t^a$$

$$dV_t = \kappa(\theta - V_t)dt + \sigma\sqrt{V_t}\,dW_t^b$$

$$\log(S_{t+1}) = \log(S_t) + (\mu - \frac{1}{2}V_t^+)\Delta t + \sqrt{V_t^+}\sqrt{\Delta t}\,\Delta W^S$$

$$V_{t+1} = V_t + \kappa(\theta - V_t^+) + \sigma\sqrt{V_t^+}\sqrt{\Delta t}\,\Delta W^V$$

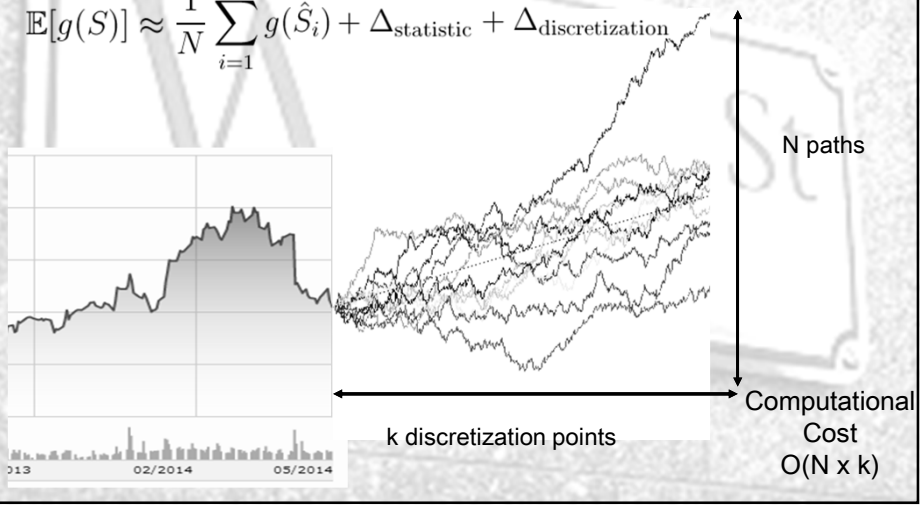
Average Profit: 4.02

$45.23

Complexity Analysis in MC

Price calculation: sample mean of simulated instances of the payoff function g(S)

$$\mathbb{E}[g(S)] \approx \frac{1}{N}\sum_{i=1}^{N} g(\hat{S}_i) + \Delta_{\text{statistic}} + \Delta_{\text{discretization}}$$



N paths

k discretization points

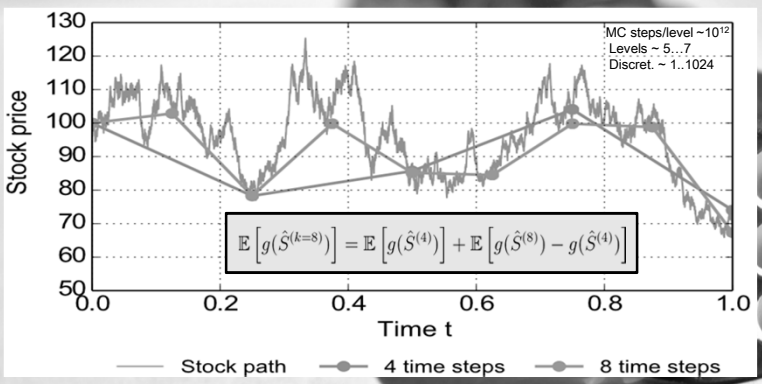
Computational Cost O(N x k)

Multilevel Monte Carlo (2008)

Sequence of MC simulations with *different discretization* and *number of paths*

- Trading-off statistic and discretization error on various *levels*
- Lower levels: coarse discretization, large variance ⇒ **large** N of low complexity
- Higher levels: finer discretization, small variance ⇒ **small** N

⇒ MLMC: for same accuracy **less** computational complexity as SMC



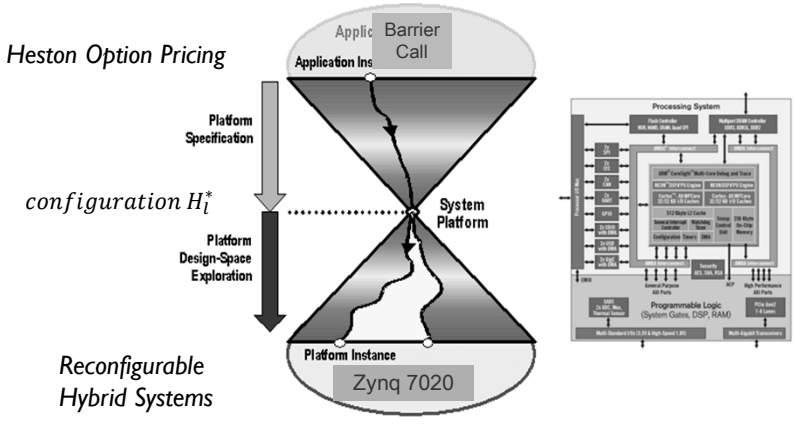
HyPER Platform for Option Pricing

MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

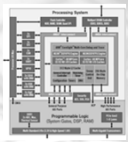
Modular pricing system for wide range of option prize calculations

Combines current trends from technology and computational stochastics

- Heston model, multilevel Monte Carlo, Xilinx Zynq platform
- New hardware oriented optimization: mixed precision

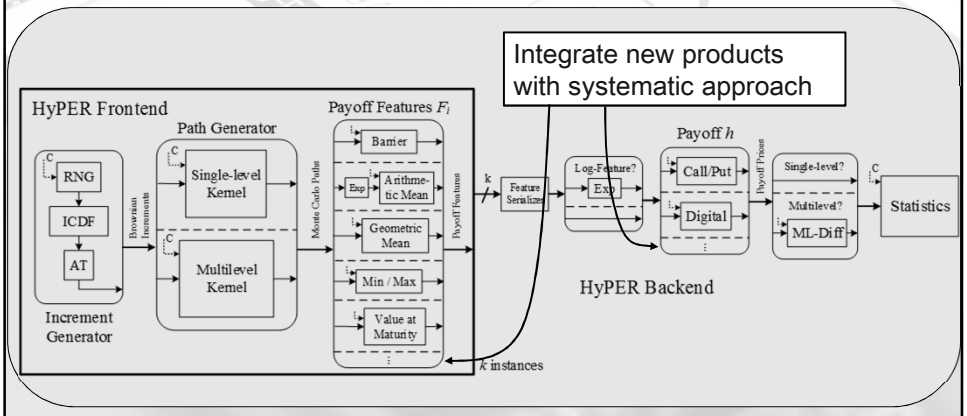


HyPER Architecture



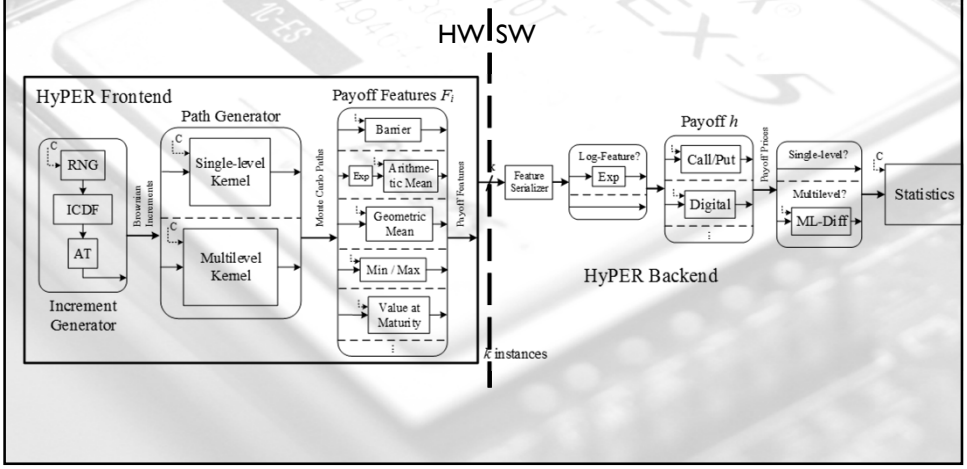
Open platform which is split into Frontend and Backend

- Frontend (HW): MC path generation & path dependent Payoff functions
- Backend (SW): path independent functions



Flexible HW/SW Split in Hyper

- MLMC level 5: features generated every 1024th clock cycle ⇒ SW
- MLMC level 1: features generated every 4th clock cycle ⇒ HW



HyPER – Static Optimizer

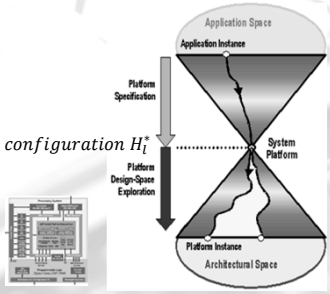
Find optimum platform configuration for given financial products and target platform

Optimization problem: maximize HyPER Performance under constraints

Choices
- Number of HyPER instances
- Number of frontends
- Position of HW/SW split
- Type of CPU/FPGA communication core

Constraints
- FPGA ressources
- CPU Load
- Bandwidth CPU/FPGA

$\Rightarrow$ ILP formulation: calculate optimum architecture H_l^* for each level



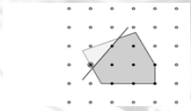
HyPER Optimization

Characterize basic building blocks

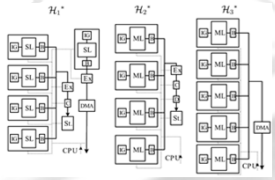
Building Blocks	LUT	FF	BRAM	DSP	CPU ns/val.
Increment Generator:					
Mersenne Twister	301	323	4	0	–
ICDF	451	592	4	1	–
Antithetic Core	228	258	0	0	–
Path Generators:					
Single-Level Kernel	4 153	4 241	2	38	–
Multilevel Kernel	5 607	5 326	6	43	–
Payoff Features F_i:					
Barrier	180	158	0	0	–
Payoff h:					
Call/Put	440	396	0	2	6
Backend:					
Feature					
Serializer $k\times1$	$30k$+65	$65k$+45	0	0	–
Exponential	900	384	0	7	250
Multilevel Difference	372	355	0	2	5
Statistics II=1	2 170	1 612	4	9	6
Statistics II=2	1 454	1 164	2	6	3

Com. Interface Ψ FPGA $\rightarrow$ CPU	LUT	FF	BRAM	Bandwidth in MB/s
Config-Bus $1\times k$	$30k$+50	$2k$+40	0	<1
Streaming-Fifo	654	611	4	20
DMA-Core	1 864	3 122	4	350

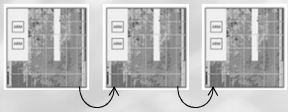
Hybrid Chip $\mathcal{F}$	LUT	FF	BRAM	DSP	ARM
Xilinx Zynq 7020	53 200	106 400	280	220	2 cores
Synthesis weight α	0.8	0.5	1	1	



Static optimization (ILP): Optimized architecture for each level



Bitstreams



Reconfigure at runtime

Mixed Precision Format in MLMC

Accuracy: accuracy of option prize algorithm

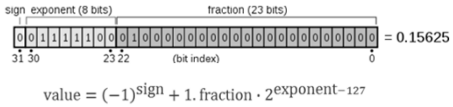
Precision: accuracy of the atomic operation

Standard approach: start with available precision of computing hardware, e.g. single/ double precision floating point. Then tune your algorithm to achieve the desired algorithm accuracy

Our approach: start with desired algorithm accuracy. Then tune computation precision to get the desired algorithm accuracy.

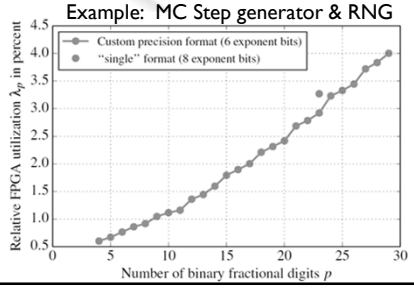
⇒ *Use customized precision for each level in MLMC*

Single-Precision



$$\text{value} = (-1)^{\text{sign}} + 1.\text{fraction} \cdot 2^{\text{exponent}-127}$$

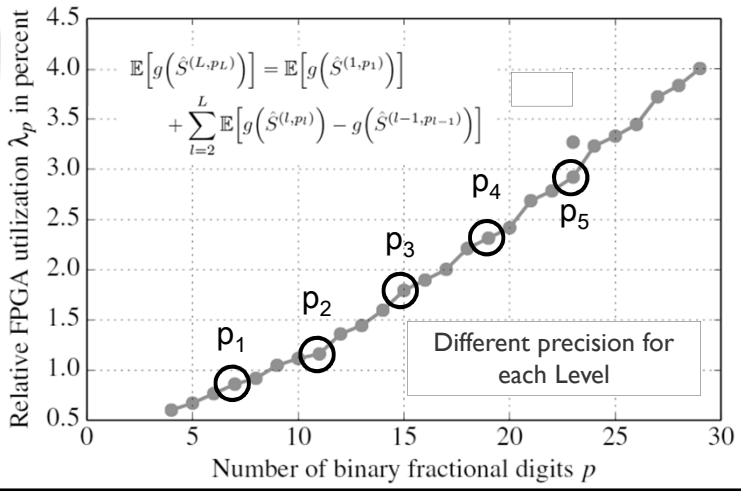
Example: MC Step generator & RNG

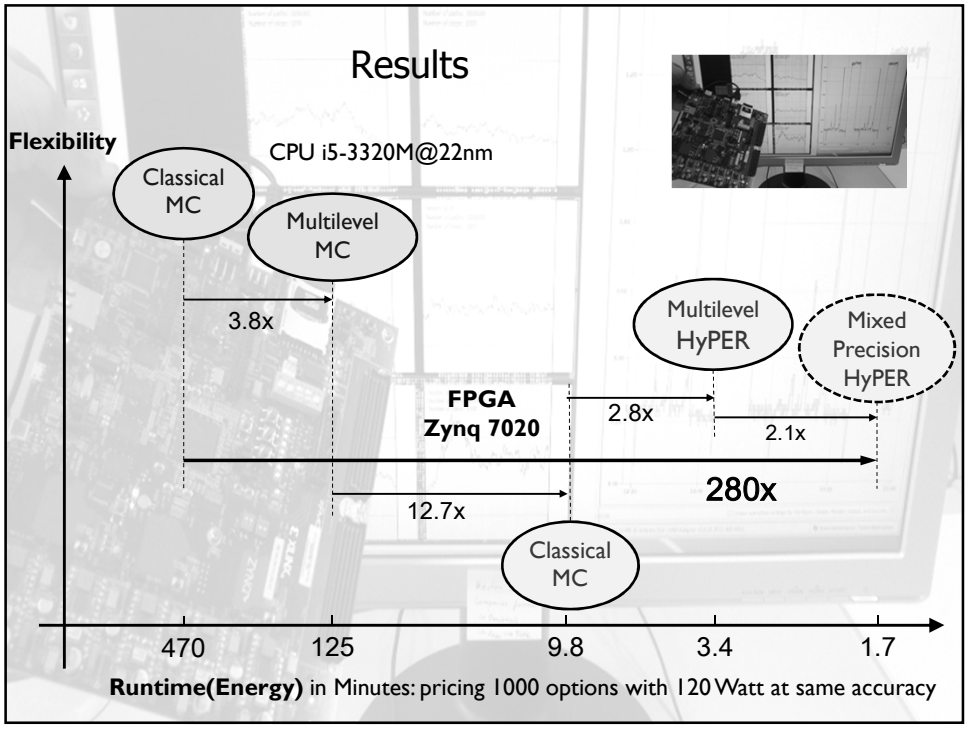


Customized Precision Results

Fast and efficient on-line algorithm to calculate precision for each level for given option prize accuracy

- Based on variance of "exact" MLMC: max 10% deviation





MICROELECTRONIC SYSTEMS DESIGN RESEARCH GROUP

For more information please visit

http://ems.eit.uni-kl.de