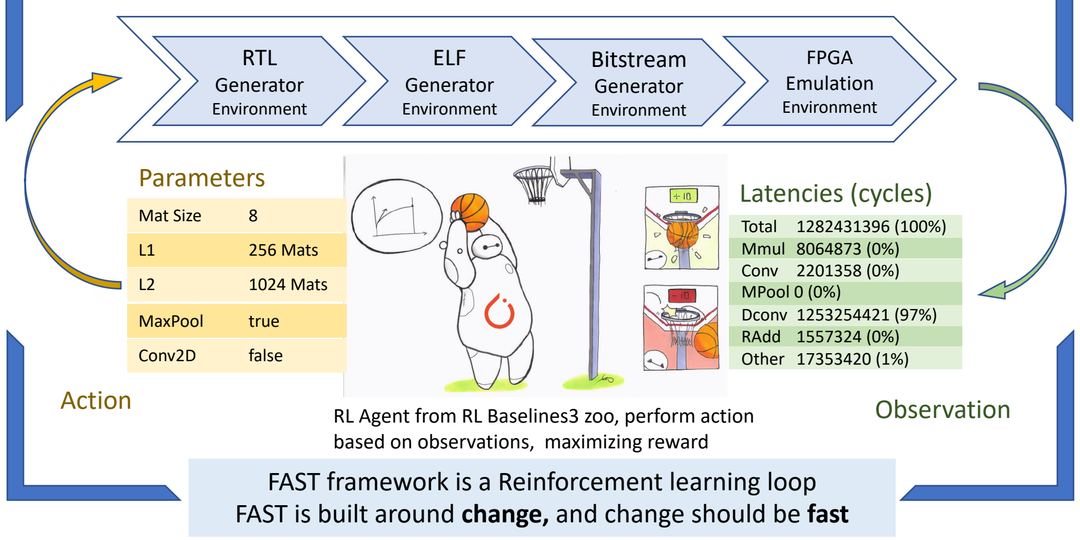
FAST: Accelerating Full-stack SoC Design-space Exploration With FPGA-in-the-Loop

Tayyeb Mahmood, Jaeyong Chung, Incheon National University, Incheon, South Korea

FAST is a work in progress concerning the Architectural and RTL **D**esign-**S**pace **E**xploration with simulation-driven ML-around approach, connecting the two ends of AIEDA's bigger picture. Open-source parameterized RTL generators like UC Berkeley's **Chipyard** and UofU's **openFPGA** have seen commercial success. To facilitate the optimization of a large design space unleashed by these projects with ML/RL- driven approaches, we present an **F**PGA-**A**ssisted **S**calable **T**est framework. FAST accelerates DSE by emulating the generated RTL and by monitoring the real-time performance and power with FPGA-in-the-Loop, to achieve an accuracy compared to conventional **model-based estimators**, at speeds many orders higher than **cycle-accurate simulators**. The key challenge to be addressed in this research is to reduce the time for RTL-to-Bitstream flow. Utilizing FAST framework, domain-experts can optimize a full-stack of AI-accelerators and reconfigurable arrays at prototyping stage or can search for an optimum design for FPGA deployments, like **Microsoft Brainwave**, and **Amazon EC2-F1** clusters.

FAST framework based on Stable-Baselines3



RL Agent from RL Baselines3 zoo, perform action based on observations, maximizing reward

FAST framework is a Reinforcement learning loop
FAST is built around **change,** and change should be **fast**

Stable-Baselines3 framework

A set of reliable implementations for Reinforcement Learning, like DQN, PPO, etc.
A rework of OpenAI baselines with a number of enhancements, based on Gym.

OpenAI Gym
Provides a standard API to communicate between RL agents and environments, and utilities to create, analyze, evaluate and vectorize the environments, action spaces and observation spaces.

RL Baselines3 Zoo
Provides framework for training and evaluating RL agents, and tuning hyperparameters
Provides a collection of tuned hyperparameters for common environments and RL algorithms, and agents.

Gym Space
A data structure to support muti-dimensional observation and action spaces, like discrete, continuous, maps and graphs.
Support random sampling of actions and validations of observations.

Gym Environment
A wrapper around a Markov decision process to learn, e.g. a retro game, a traffic controller or parameter tuner as FAST.
Supports two simple functions, reset and step.

Gym Vectorization
Provides framework to run multiple instances of Environments, in parallel (multiprocessing).
Input: a batch of actions,
Output: a batch of observations
Scalability in clusters and cloud.

FAST framework

Goal
To automate, decouple and accelerate the DSE flow from Architectural parameters to RTL to Bitstream to Execution, as shown in the figure.

Challenge
RTL to bitstream flow is optimized for a time-consuming single pass, followed by iterative optimizations. Incremental design flow only allows minor touching.

Motivation
DSE flow can take advantage of hierarchical design, with synthesis and P&R of only those modules which are modified by parameters.

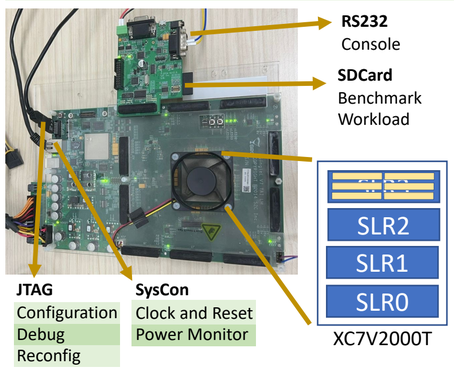
We divide time-consuming RTL-to-Bitstream pass into a sequence of decoupled subprocesses, in order to optimize each subprocess for DSE, individually.

FAST Track

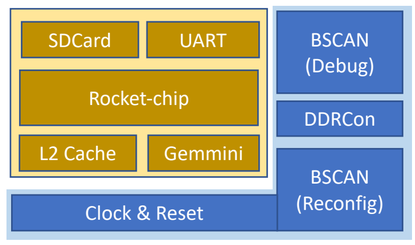
RTL generator	ChipyartFIRRTLEnv	Scala project is assembled into standalone jar to reduce FIRRTL generation time and for scalability.
	ChipyartVerilogEnv	Verilog generation time can be reduced by differential conversion.
ELF generator	WorkloadGenEnv	c/c++ Compilation is fast
	WorkloadDeployEnv	ELFs must be prestored on SDCard, currently it is manual process.
Bitstream generator	FPGASynthesisEnv	Full synthesis consumes more than 30 minutes, partial synthesis consumes 10 minutes.
	FPGAPlaceEnv	Full Placement consumes 20 minutes, partial placement takes 10 minutes.
	FPGARouteEnv	Full pouting consumes more than 40 minutes, partial routing consumes 10 minutes.
	BitstreamGenEnv	Full bitstream generation consumes more than 5 minutes, partial bitstream consumes < 1 minute.
FPGA emulation	BitstreamConfigEnv	Configuration is fast.
	FPGAEmulationEnv	Less than 7 minutes for Resnet50, however most of time is consumed in loading image from SDCard.
	ObservationEnv	Scanning data from logs is fast.

FPGA Platform

SingleE V7 Logic Module
Xilinx XC7V2000T-FLG1925
2.5D SSI, 4 SLRs (24 Clock Regions)
RS232 and SDCard Add-on module
8GB DDR3 SODIMM

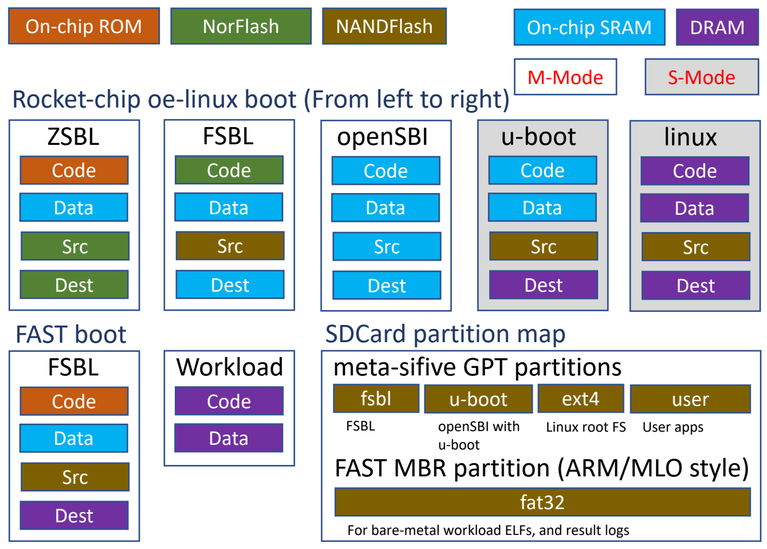


RTL Test Harness- Chipyard



Shell: Sifive's fpga-shells framework
BSCAN: Xilinx Primitive instance
DDRCon: Xilinx DDR3 MIG
SDCard: Sifive's sifive-blocks SPI
UART: Sifive's sifive-blocks UART
Rocket-chip: UCBAR's RISCV core
L2Cache: Sifive's inclusive L2 cache
Gemmini: UCBARs systolic array for MatMul, accelerator units for conv, relu and maxpool.
Clock & Reset: 200MHz master clock and a global reset from SysCon

FAST Software execution order



FAST First Stage Boot Loader runs from FPGA BlockRAM
FSBL loads Bare-metal executable ELFs directly from SDCard
FAST partition scheme is simply FAT32, compared to complex partition structure. This allows a batch of workloads to be executed, their performance logs saved.

FAST FSBL execution

POR init
UART/SDCard init
Read ELF from SDCard
Copy text segment
Copy data segment
Init BSS
Jump to text entry

FAST Workload structure

Sifive's freedom-e-sdk main
UCBAR's Gemmini-ROCC-TESTS overlays

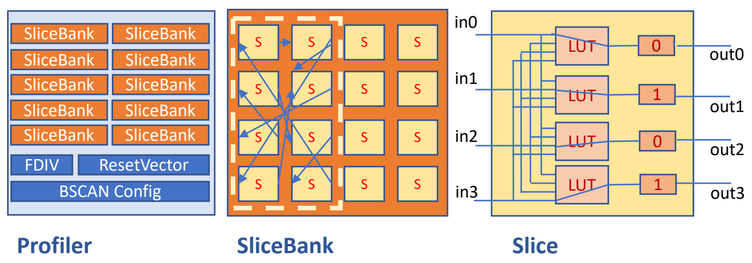
Micro-Kernels
MatMul, Conv, etc.

ImageNet
ResNet50
Mobilenet
AlexNet

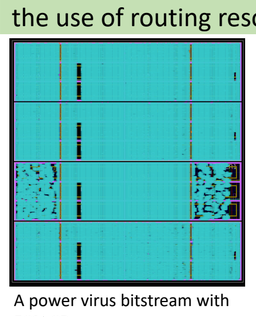
MPLS

Power profiler

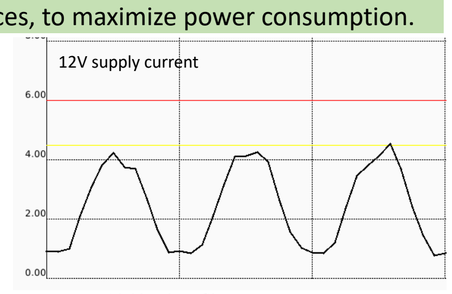
A parameterized Chisel generator to stimulate FPGA activity
Generates bitstreams with Sized Clock Region activities.
- Scalable clock region occupancy
- Dynamic frequency scaling
- Pulse Width Modulation based load profiling



Profiler has slicebanks, one for each clock region.
FDIV is a counter for divided clocks, one of which is MUXed out.
ResetVector is a shift register, clocked with FDIV output.
Each bit of ResetVector drives a Slicebank reset.
BSCAN config is a capture register on FPGA JTAG tunnel.
Config register drives FDIV MUX and ResetVector pattern.
SliceBank has a number of slice groups.
A slice group is a group of slices, the even outputs of whom are randomly connected to the odd inputs, and vice versa.
Slice models an FPGA slice, the registers of which are reseted to '0101….' pattern, each LUT passing one of input to a register.
The even-odd connections ensure that the internal node have maximum activity at each clock edge, and randomization ensures the use of routing resources, to maximize power consumption.



A power virus bitstream with 50% CR occupancy



Trapezoidal load profile

Conclusion

We note that partial reconfiguration significantly reduces Bitstream generation time, however total test time still exceeds 30 minutes. Although it is an order better than cycle-accurate simulators for designs of this complexity, we can further reduce this time with techniques such as differential generation and synthesis, using a netlist cache and using RapidWright framework for data driven placement and routing. FAST environments can be used for DSE and policy optimization where a policy may be a sequence of parameters with minimum changes at each step, like Gray coding.