Hybrid Hardware/Software Synergism

By
Patrick W. Burke
Philip Sweany
University of North Texas, Denton, TX

Agenda

- Introduction
- History
- Problem Statement
- Solution
- Benefits
- Future Work
- Conclusion

Introduction

- Who am I to say
  - US Air Force Academy – BSCS (1977)
  - 30 years in Software in Military-Industrial Complex
  - University of North Texas – MSCS (1988)
  - IEEE Certified Software Development Professional
    - 2002 -- First class of recipients
  - Retired last year
  - University of North Texas PhD Student – 9 years

History

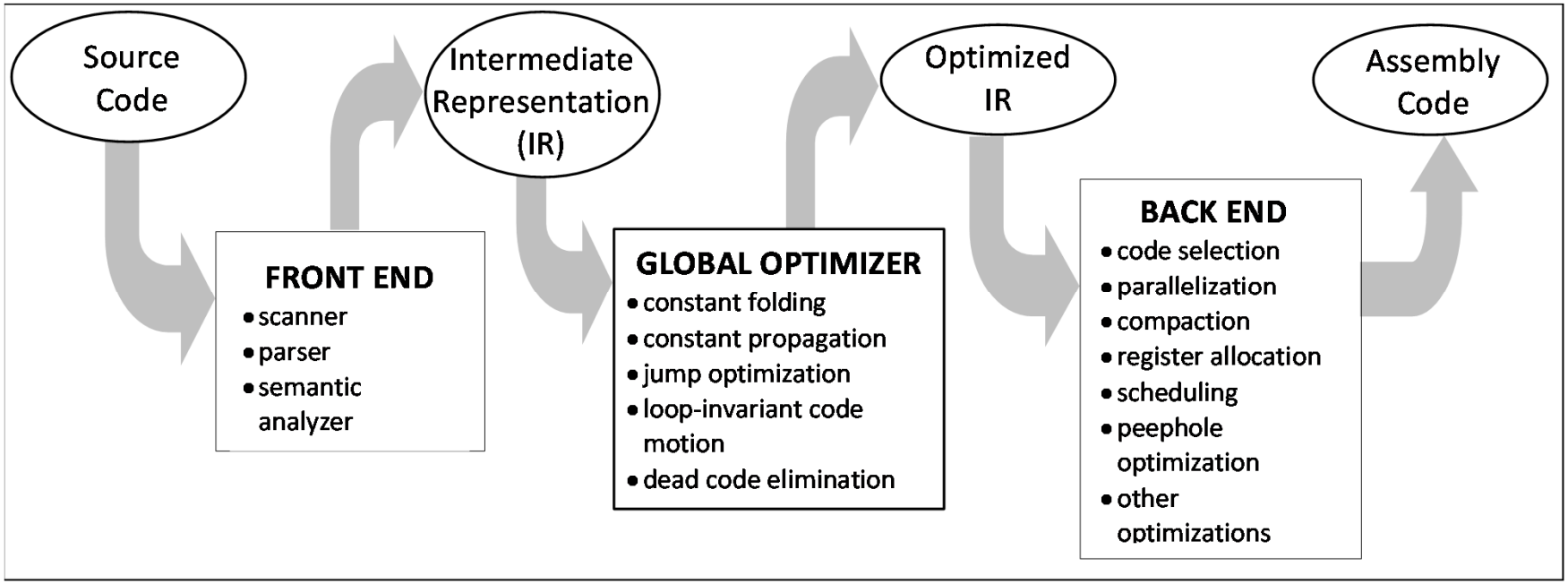
- Software Engineering still not very good at what we do
  - Far too many software failures
  - Unreliable predictions of schedule/cost

- Notable Software Failures
  - Y2K (19?? – 2000)
  - Mars Global Surveyor (2006)
  - Automatic Brokerage problems on Wall Street (2012)

History (cont.)

- Focusing on processors
  - Moore's Law vs Physics
    - Limit to transistor density?
      - More processors = more heat to dissipate
    - Processor clock speeds have stabilized
    - Current market demand for smaller footprint / less power consumption
  - Multicore processor issues
    - Communication between processors
    - Efficient utilization of processor capacity?

Typical Compiler



- Generally tuned to a single processor or multi-core processor
  - Multi-core utilization is still low
  - Parallelism generally manually extracted in design (threading)
  - Threaded-processing can help … or hurt

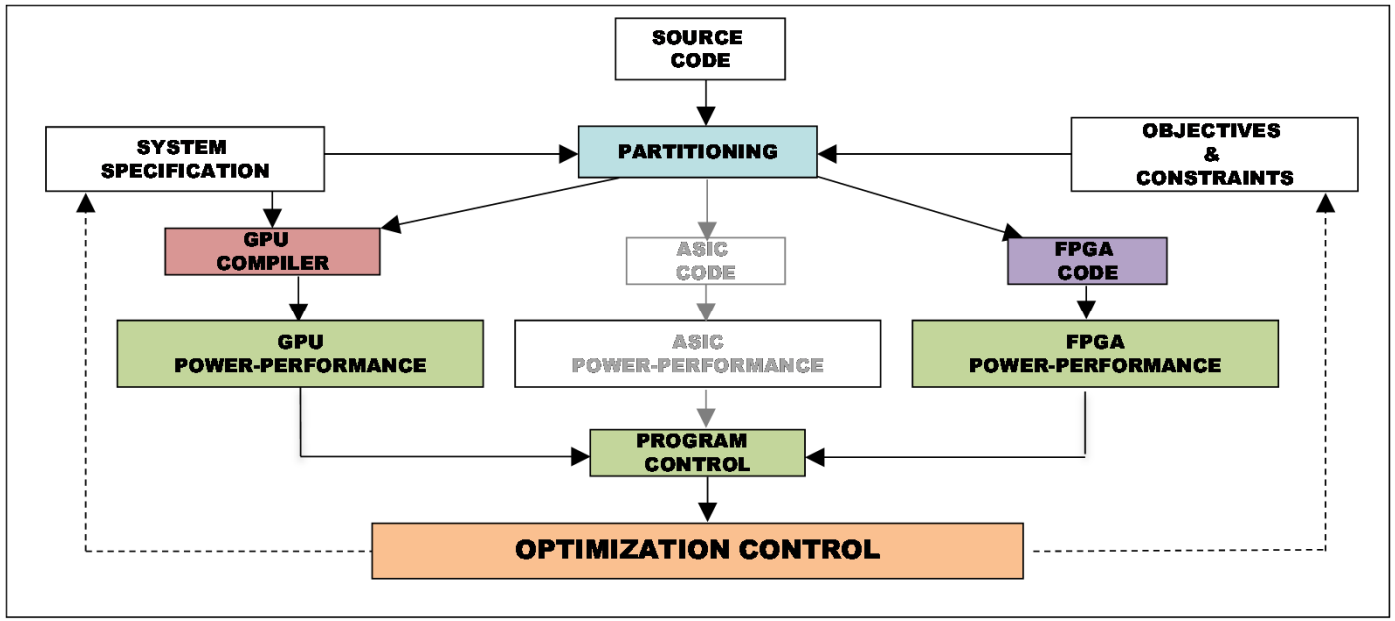
Problem Statement

- How can we squeeze more processing, using less power (generating less heat) out of modern computing devices for hand-held technologies.

Solution

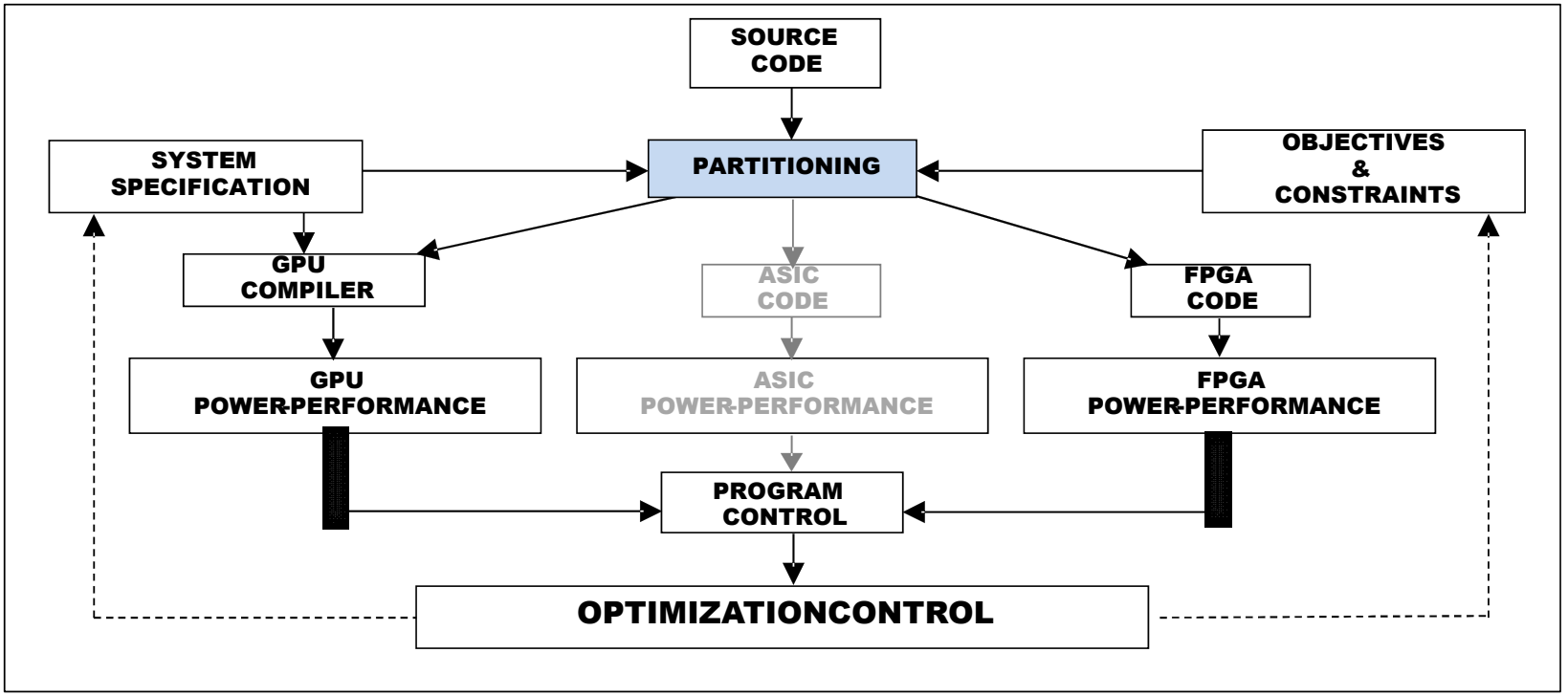
- Heterogeneous Systems on a Chip
  - Provide a variety of processors with different processing/power/heat characteristics
    - GPU (General Processing Unit)
    - DSP (Digital Signal Processor)
    - FPGA (Field-Programmable Gate Array)
    - ASIC (Application-Specific Integrated Circuit)
  - Spread processing across the components in as “economical” fashion as possible
  - Evaluate the mix of components, if warranted, to achieve better results

Hy-C (Hybrid Compiler)



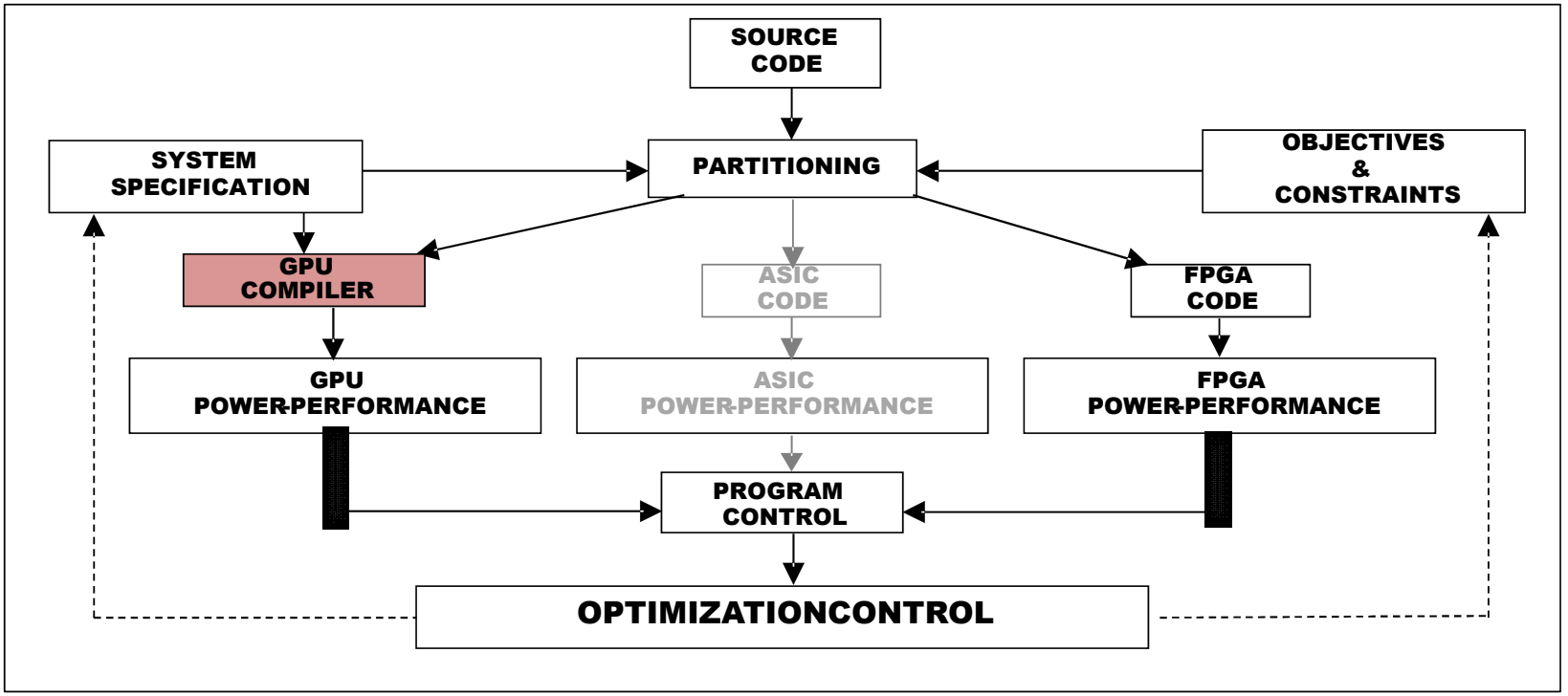
- Retargetable compiler (via System Specification)
- Automatically extracts parallelism (Partitioning)
- Result include performance parameters (generally an estimation of processing time and heat/power results) in accordance with Objectives & Constraints

Hy-C (Hybrid Compiler)



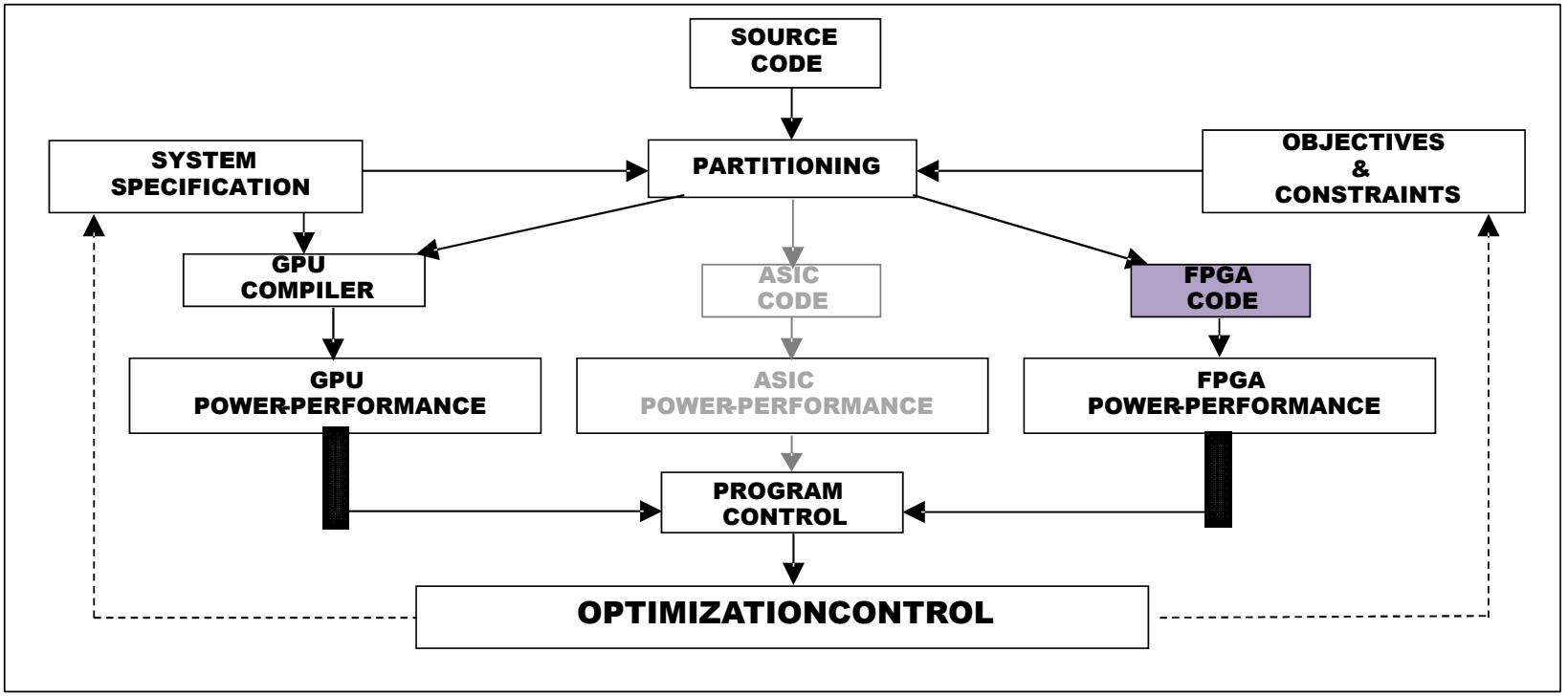
The Partitioning Function uses subgraph matching to assign blocks of code to the most efficient processing component. It also creates the synchronization protocol PROGRAM CONTROL (queues, semaphores, signals, etc.) to ensure proper sequencing of the code execution.

Hy-C (Hybrid Compiler)



The GPU Compiler(s) is/are the native compiler(s) for the selected General Processing Unit components.

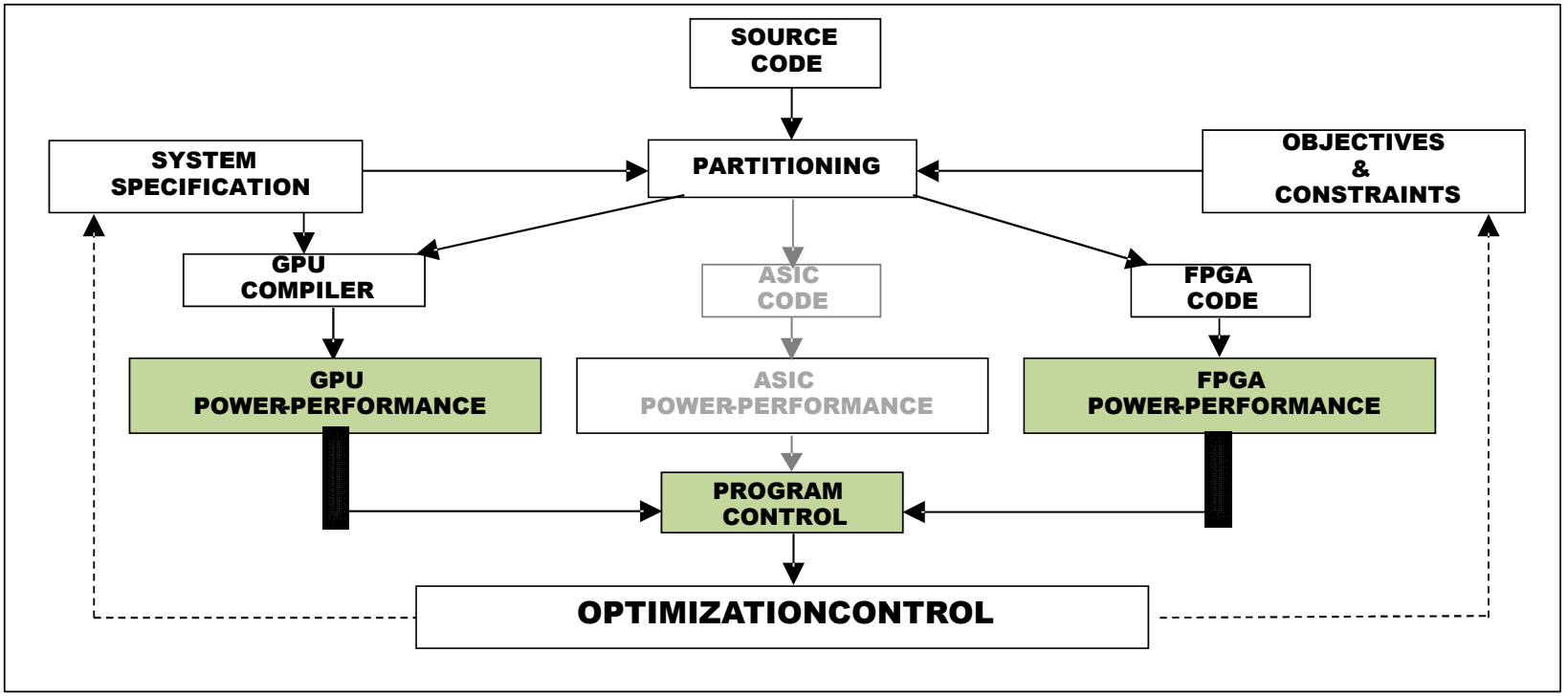
Hy-C (Hybrid Compiler)



The FPGA Code Generator translates Intermediate Representation directly into FPGA code.

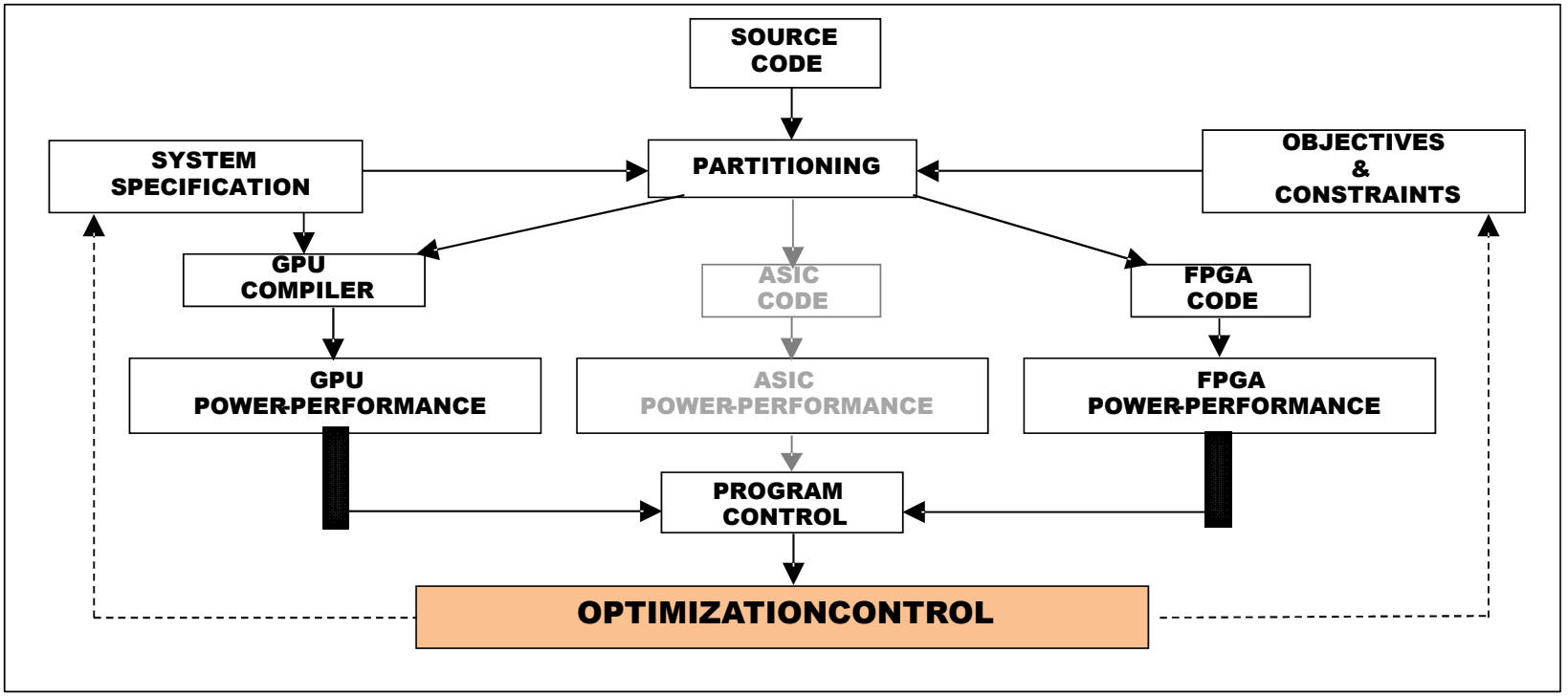
***** FPGA Code which is found to have high utilization should be considered for use in an ASIC**

Hy-C (Hybrid Compiler)



These functions simulate the PERFORMANCE (processing time, energy/heat characteristics) of the code on the associated components , based on performance specifications provided in the System Specification and the Objectives and Constraints.

Hy-C (Hybrid Compiler)



The Optimization Control is currently planned as a manual function to drive modification of the Objectives or System Specification. In the future, Analysis of the performance and power/heat statistics might also suggest a different hardware configuration to possibly yield better performance.

Benefits

- Excellent tool for **Hardware/Software Co-Design**
- Assignment of code blocks to the most efficient hardware available will
  - Allow tradeoff decisions (cost vs performance)
    - Should a different mix of hardware be used?
  - Yield synergistic execution times
    - As fast as, or faster than, any hardware component alone
      - Depending on the type of processing being performed

Future Work

- Current vision works with static hardware configuration

- Automation of the Optimization Control function could automate part of the **Hardware/Software Co-Design** process to select the ideal hardware for given source code

Conclusion

- We anticipate the use of in the **Hardware/Software Co-Design** process will benefit from the Hy-C Retargetable Compiler by providing optimum code execution results (minimal processing time, heat generation, and power consumption) for a given configuration of hardware components.
  - This will yield a much better execution of the code when compared to homogeneous processor sets.
- Follow-on work could automate the selection of hardware to result in the “best” execution of the given source code.

ACRONYMS

- ASIC Application-Specific Integrated Circuit
- DSP Digital Signal Processor
- FPGA Field-Programmable Gate Array
- GPU General Processing Unit
- Hy-C Hybrid Compiler