Transparent Hardware Generation from Assembly Code at Program Execution Time

- Embedded scenarios demand ever increasing computational performance and power efficiency
- Deployment of ASICs proves inadequate
  - Low flexibility/reprogrammability
  - Long development time and cost

- Another strategy:
  - Augmenting General Purpose Processors with co-processors which explore parallelism

- **Dynamic Hardware/Software Partitioning**
  - Detecting and migrating execution of demanding software kernels to dedicated hardware

- Proposed approach:
  - Mapping loops running on a microprocessor to a **Reconfigurable Processing Unit (RPU)**
  - Transparent to host processor
  - Transparent to the application programmer

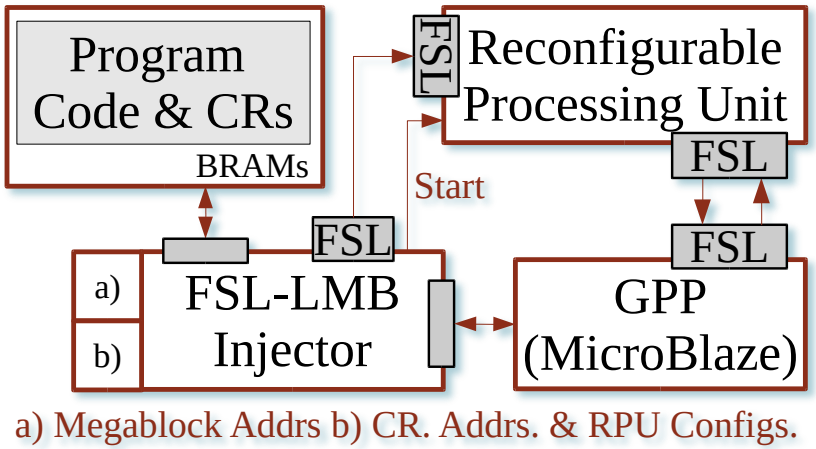

Fig.1 – GPP augmented with an RPU. Injector module performs some HW/SW partitioning steps. Local memories contain instructions that perform synchronization between RPU and CPU.

Main Objectives/Tasks

- Transparent runtime HW/SW partitioning
- Develop flexible RPU structure
  - With memory access support
  - Viable for reconfiguration by partial bitstreams
- Runtime generation of partial bitstreams
- Validation with real-world applications

Description

- Modern embedded systems must handle computationally intensive applications under severe resource and power constraints. FPGAs can be used to provide specialized accelerators for well-identified tasks in embedded systems, but this still requires a large effort for hardware development and application adaptation.

- This work will design a method for creating reconfigurable hardware modules from running assembly code in a transparent manner. The work will concentrate on synthesis of the hardware accelerators, and on mechanisms for transparent activation during application execution. Hardware accelerators will be generated on-system and loaded by dynamic reconfiguration; support for multiple application loops can be combined in one accelerator. The new modules will be used transparently by the application, with all communication, including data memory accesses, handled by the underlying infrastructure