

思源科技—台灣大學合作研究計畫

電子系統階層之正規模型與
最佳化研究

Formal Modeling and Verification of Electronics
System Level Designs

主 持 人：黃鐘揚 教授

執行單位：台大慶齡工業研究中心

執行期限：2007/03/01 ~ 2008/2/29

計畫聯絡人：黃鐘揚 電話：02-3366-3644
傳 真：02-2367-1909 E-mail：ric@cc.ee.ntu.edu.tw

一、導論

With the increasing complexity of modern System on Chip (SoC) designs, Electronics System Level (ESL) design methodology is becoming an inevitable trend.

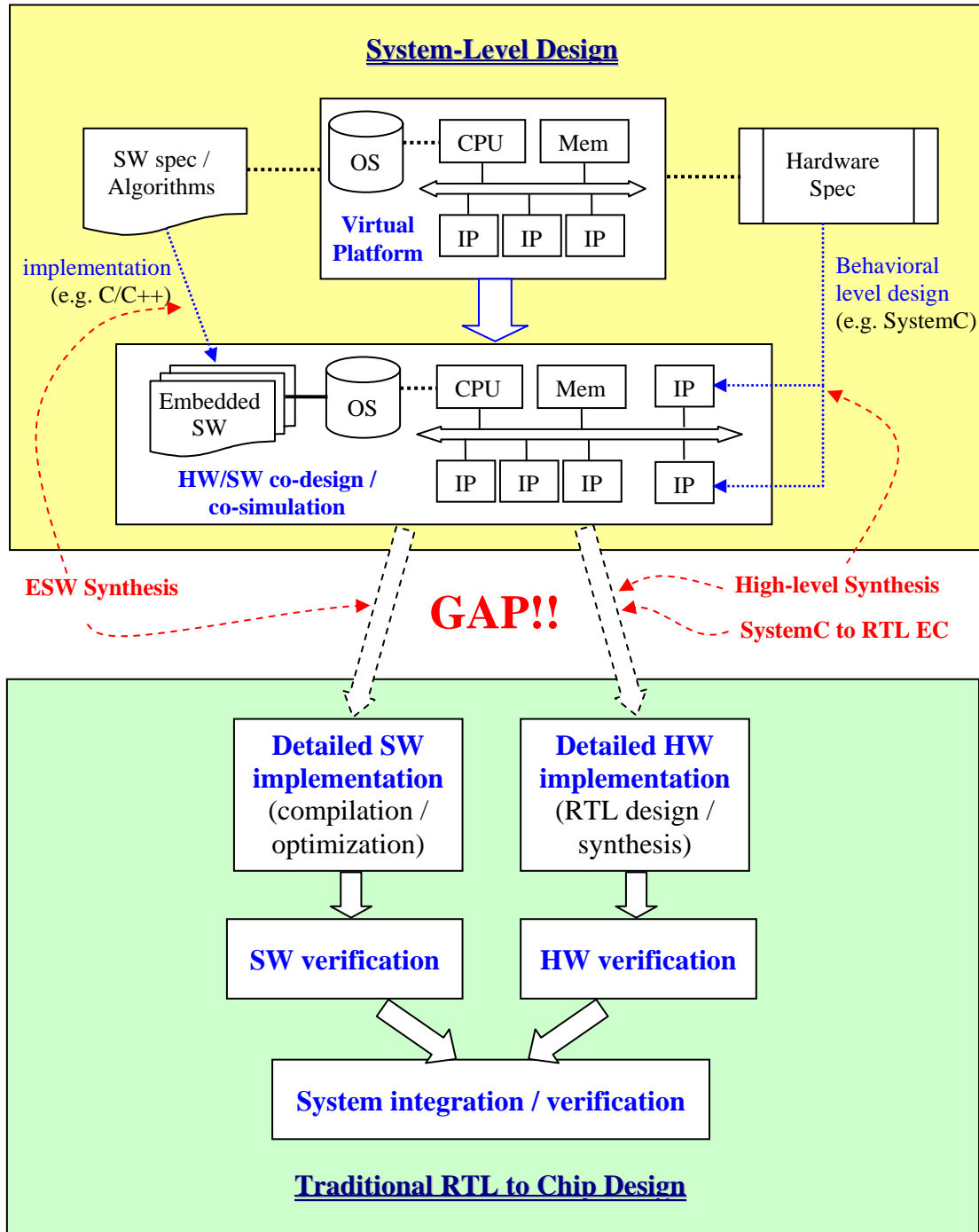


Fig. 1 ESL Design Flow

The typical ESL design flow is as shown in Figure 1 — it can be divided into two phases: first, the “system-level design” phase implements the system prototype in

certain high-level abstraction languages such as SystemC or C/C++. It is usually implemented on a “virtual platform” where the target processor core(s), bus architecture, hardware IPs (Intellectual Properties), and even Operating System (OS) are pre-determined and provided. Consequently, we can design the hardware (HW) / software (SW) modules on top of it and therefore perform co-simulation in this early design stage. The correctness of the system-level prototype should also be verified.

The second phase of the ESL design flow, on the other hand, is the traditional “RTL to chip design” phase. The hardware components (IPs) are implemented in certain HDL languages such as Verilog or VHDL, and the software are compiled into processor-dependent environment. The design tools and methodologies in this phase are relatively much more mature as they have been developed and widely used for more than one decade. The individual components can thus be highly optimized and later integrated as a complete SoC.

Although these two phases of the above ESL design flow seem work well individually, there is a big gap between them — the transition from the system-level to the RTL designs still requires highly manual efforts. From the hardware side, the support of the system to RTL synthesis is still limited and therefore the designers need to re-implement the detailed RTL designs on their own. As a result, the equivalence of the system and RTL designs can not be guaranteed and thus the verification efforts spent in the system level cannot be re-applied in RTL.

On the other hand, for the system software to work correctly on the target SoC environment, we need to consider many factors such as the hardware constraints (e.g. processor speed, power consumption, etc), real-time scheduling, OS and firmware dependency, and so on. However, currently there is no tool available (or say good enough) for this embedded software synthesis and optimization purpose. As a result, we need to spend a lot of efforts in implementing the embedded software and make it work on the SoC platform.

In this proposal, we aim to bridge the gap between system and RTL design flows. Specifically, we will focus on the formal modeling and verification techniques of the system level designs. Our objective is to first devise a “formal representation” of the system-level designs so that we can later apply various logic and arithmetic techniques to formally verify the correctness of the design. The formal representation here refers to a data structure that (1) is a graphical representation of the hardware and

software control and data flows, (2) can formally capture the semantics of the hardware and software components, (3) is flexible enough to transform between different abstraction levels, (4) can perform optimization on varied design constraints, and (5) can be mapped back to system-level design language such as SystemC. Based on this formal representation of the system level designs, we will be able to develop the engines for the system to RTL formal verification.

We plan to conduct the research for the above objectives in the first year. With this formal model of the system level designs ready, it will be easier for us to conduct further researches on areas such as “system-level spec validation”, “system-to-RTL hardware equivalence checking”, and “embedded software synthesis”, etc.