Rapid Prototyping of Application-Specific Signal Processors: Educator/Facilitator Current Practice (1993) Model and Challenges

Vijay K. Madisetti [1], *Jack Corley* [2], *Gary A Shaw* [3]
[1] ECE - Georgia Tech, Atlanta, GA 30332-0250
[2] ATG - SCRA, N. Charleston, SC 29418
[3] MIT Lincoln Laboratory, Lexington, MA 02173-9108

1

Abstract

The Rapid Prototyping of Application-Specific Signal Processors (RASSP) project of the US Department of Defense (ARPA and Tri-Services) targets a 4X improvement in the design, prototyping, manufacturing, and support processes (relative to current practice). The authors present a "current practice (circa 1993)" model for the design and prototyping of application-specific signal processors developed as part of the RASSP Educator/Facilitator (E/F) Program. A number of limitations in current design practice are highlighted together with challenges faced by candidate solutions. The E/F Program proposes that this model be used as a baseline in the evaluation of newer RASSP prototyping methodologies and processes.

1 Introduction

In this section we introduce classes of high-performance application-specific parallel processors that are the subject of the Educator/Facilitator current practice (1993) study. In Section 2 we present "current practice (circa 1993)" based on extensive study of industrial practice first-hand and through communications with various industrial and defense contractors at Lockheed Sanders, Martin Marietta, Hughes Avionics Systems, Motorola, MIT Lincoln Laboratory, and Raytheon corporations, by the authors as part of the RASSP Education/Facilitation and Benchmarking efforts over the past year. However, the viewpoints presented in this paper are solely those of the authors, and not necessarily of these organizations. Section 3 describes the challenges facing application-specific parallel processor specification, implementation, verification and support, that the RASSP program attempts to address in the near future.

1.1 Representative Architecture

A typical high-performance avionics parallel signal processor operation flow is shown in Figure 1 [3]. The inputs are recorded by sensor arrays, and the data is pre-processed by an array of (typically hardwired) computational elements, comprising the *sensor-specific processing (SSP)*, that are optimized with the sensor array and the recording environment. Typical SSP operations include range adjustment, background subtraction, and matched filtering. Given the high computational throughput and restricted functionality, and severe form constraints (size, volume, area and power), the SSP functions are typically ASICs with non-standard interfaces. SSP functions are also referred to as *time-dependent* processing.

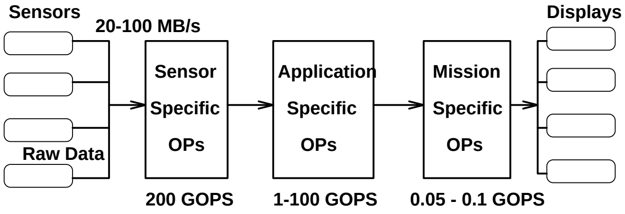


Fig 1. **Typical System Architecture.**

After this time-critical processing is completed, the application-specific (ASP) parallel processing (about 30-100 processors) is commenced on an array of processors and communications elements, with appropriate test, control, and maintenance structures. Typical ASP operations include coordinate transformation, track-to-track correlation, Kalman filtering, tracking, and parametric estimation and involve application related functionality. The ASP functions also require relatively high throughput, and it is desired that they have as much flexibility (i.e., programmability) as possible, together with certain form factors. In an ASP implementation lie the multi-objective function optimization and tradeoffs among form factors, performance, programmability, ease of upgrades, and capability for test and diagnostics. ASP functions can also be referred to as *object-dependent* processing.

The mission-specific (MSP) processing typically requires interpretation of the ASP processing, and can be confined to a few processors that are often co-located

---

[1] This research has been supported by the Advanced Research Projects Agency (ARPA), Electronic Systems Technology Office (ESTO) as part of the RASSP program.

within the ASP box. These functions include clutter analysis, track handoff, decision analysis, kill assessment, etc. The relative orders of processing in GOPS are also depicted (decreasing to the right) in Figure 1; while memory requirements typically increase towards the right, with the most memory required by the MSP (typically 100-400 Mbytes) stage (because decisions could be made over multiple frames). Typical form factor constraints for volume, power, weight, and I/O rates are in the order of 2-10 cuf, 40-500W, 10-60 lbs, and 4-30 Mbytes/second, while for low-end low power portable applications they are considerably more severe (in size and power). Interprocessor communication bandwidth requirements can range between 40-1000 Mbytes/second.

1.2 Architectural Design Space

A combinatorially significant number of alternatives exist in the implementation of the SSP, ASP, and MSP functionality, and a few key architectural attributes [3] are listed below:
Computational Elements - types (data, control, ASICs, or DSP) of processors, coarse or fine-grain task assignment, heterogeneous or homogeneous processing, size of memory elements, degree of coupling between memory and processors, software and algorithms utilized, SIMD or MIMD types of control.
Communication Elements - buses used, backplane architectures, interface to buses within system, interfaces to environment, routing and communications protocols.
Topologies - allocation of physical resources (number of processors, memories, communication elements), interconnection topologies (ring, bus, etc) and technologies (fiber, parallel/serial, etc), I/O configurations, integrated test and fault-tolerance capabilities.

1.3 Typical System Specifications

A number of specifications/requirements form the input to the prototyping process (and they can be weighted in relative order of importance). These include, in no specific order: (1) Functionality and Performance, (2) Environment, (3) Interfaces and Packaging, (4) Security, (5) Schedule for deployment, (6) Cost, (7) Software and hardware restrictions, (8) Size and volume, (9) Weight, (10) Power, (11) Reliability, (12) Maintainability, (13) Fault-tolerance, (14) Scalability, (15) Standardization. A successful design has to achieve a satisfactory degree of compliance with each of these specifications [3].

2 Current Prototyping Practice

We now examine current practice (1993) in the prototyping of application-specific parallel signal processors.

2.1 System Development Phases

The lifecycle of a typical large system roughly follows the six phases shown in Figure 2. The focus of

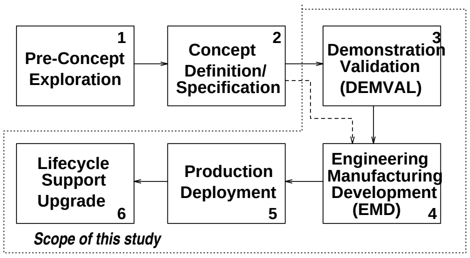


Fig 2. **Typical Signal Processor Development Phases**

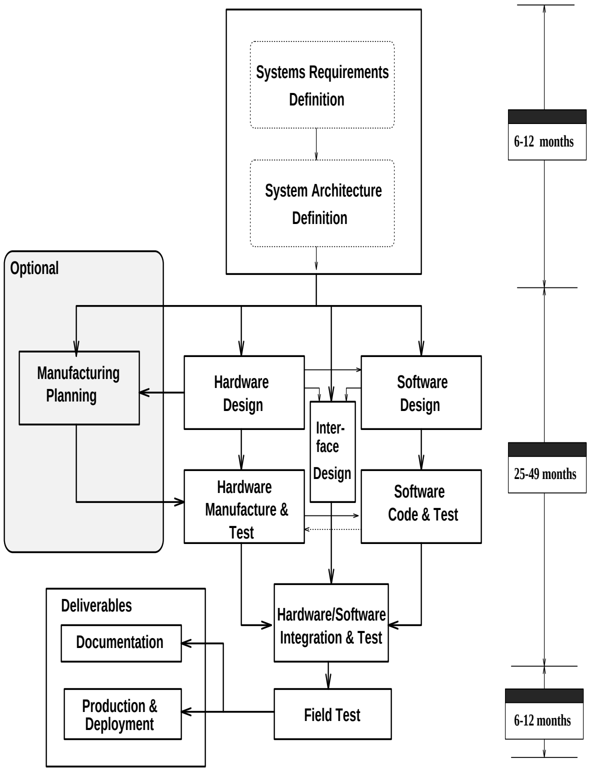


Fig 3. **E/F "Current Practice (1993)" Signal Processor Design Process.**

this study is on phases $3-6$. Often, phase 3 is bypassed, and the sequence shown by the dotted-line is followed. In some cases, phase 3 follows 4, and is followed by phases 5 and 6. The dollar cost figures for each of these phases in terms of resources (capital and personnel) can run into few tens of millions in the initial phases of Figure 2, and as high as a few hundreds of millions (or more) in the later phases of system development, deployment and maintenance. The ratio of cost incurred/committed rises steeply with the onset of phase 3. Clearly, decisions and tradeoffs made in the early phases have significant financial impact later in the system development lifespan. In addition, the long time span (often $8-10$ years) between phases 2 and 5 render some aspects of the technology obsolete by the time a concept is fielded. While customer input is significant in phases $1-3$, it diminishes in phases $4-6$ leading to significant risks to the manufacturer in fixed-price contracts, or to the customer via cost overruns when redesign or rework is required. The lifecycle support (phase 6) for the deployed system can last between $10-30$ years. The capability for rapid upgrade is an effective insurance against technological obsolescence of the fielded system [7].

3 E/F Current Practice (Circa 1993)

Figure 3 presents the E/F current practice model for system design. The various stages in a "waterfall"-type process flow are demarcated together with time ranges (min, max) for each stage. Figures 4, 5, 6, and 7 describe each of the stages of Figure 3 in greater detail, again presenting details of the time required for each substage, together with tools used, process inputs and outputs. Because the figures are very detailed, it is hoped that they are self-explanatory. The time lines have also been validated via communications with the industrial entities involved in large system design and implementations.

3.1 Areas for Improvement

Some observations may be made at this point with respect to the design flow in Figure 3 that provide pointers to areas for improvement [7].

- *Automation and Enterprise Integration* — Though progress in automation has been significant in the area of Hardware Design and to a lesser extent in Software Design, very little impact has been felt on system-level design, architectural exploration and tradeoffs, hardware/software integration, integrating manufacturing and product design activities, and reuse database libraries. Most of the information transferred between various stages of the design process is manual and usually documented on paper with little standardization. It may be argued, for instance, that algorithms designed in a high-level graphical environment can be translated to lower-level hardware and software implementation through code-generation mechanisms. However, these approaches are seldom efficient (unless optimized and

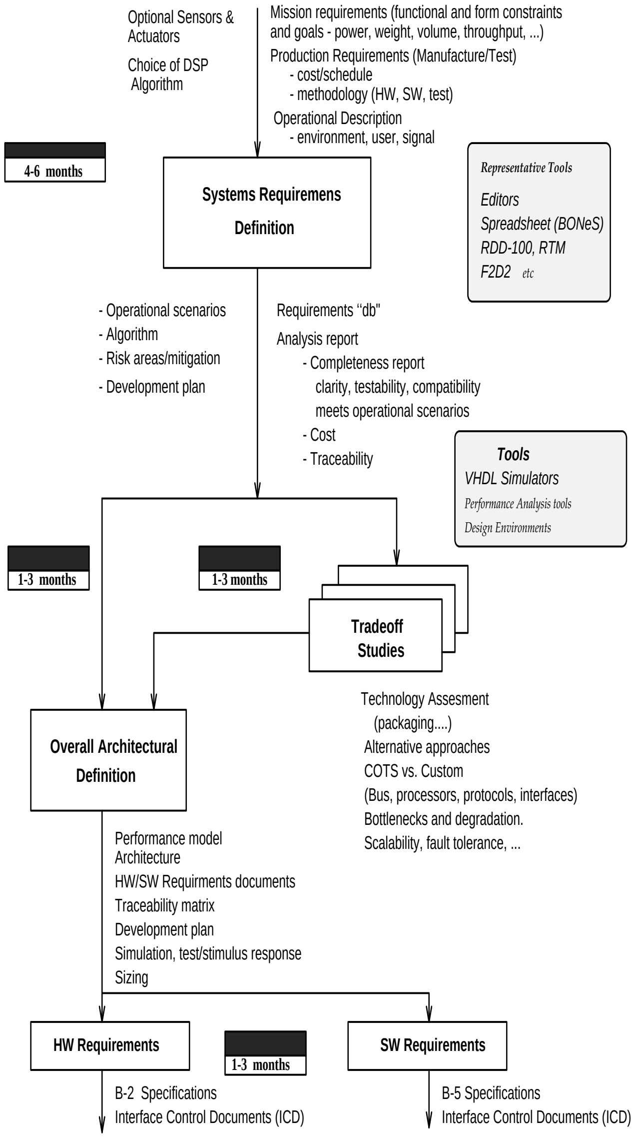
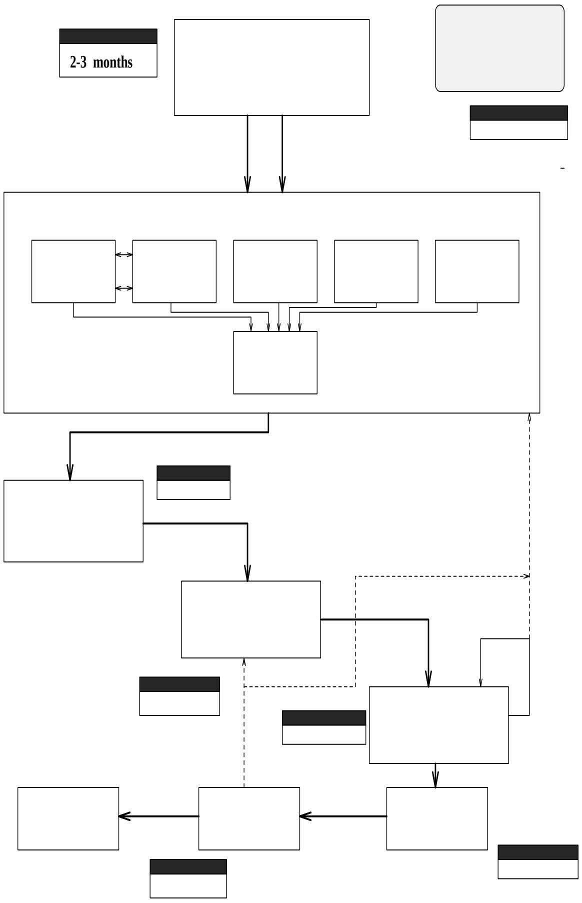


Fig 4. **System Requirements/Architecture Definition (1993)**


2-3 months