8. Real-time Performances: Algorithms vs Architectures

8.1 Objectives

- To evaluate real-time performance through case studies.
- To explore the matching and mismatching of algorithms with architectures.
- To identify the characteristics of architectures based on their real-time performance.

8.2 Introduction

The performance demands in modern engineering applications have motivated the utilisation of complex and computationally intensive algorithms, including adaptive and intelligent control methodologies. This in turn has led to hard constraints on computing requirements of the hardware, to achieve the required sample times. Therefore, real-time performance in control applications where the use of advanced control methods is warranted becomes difficult to accomplish. Many demanding complex control processes cannot be satisfactorily realised with conventional uniprocessor and multiprocessor systems. Alternative strategies where multiprocessor-based systems are employed, utilising high-performance processors and parallel processing techniques, could provide suitable methodologies (Jones, 1989; Tokhi et al., 1992).

In a conventional parallel system all the PEs are identical. This architecture can be described as homogeneous. However, many signal processing and control algorithms are heterogeneous, as they usually have varying computational requirements. The implementation of an algorithm on a homogeneous architecture is constraining and can lead to inefficiencies because of the mismatch between hardware requirements and hardware resources. In contrast, a heterogeneous architecture having PEs of different types and features can provide a closer match with the varying hardware requirements and, thus, lead to performance enhancement. However, the relationship between algorithms and heterogeneous
architectures for real-time signal processing and control is not clearly understood. The mapping of algorithms onto heterogeneous architectures is, therefore, especially challenging. To exploit the heterogeneous nature of the hardware it is required to identify the heterogeneity of the algorithm so that a close match is forged with the hardware resources available (Baxter et al., 1994).

One of the challenging aspects of parallel processing, compared to sequential processing, is how to distribute the computational load across the PEs. This requires consideration of a number of issues, including the choice of algorithm, the choice of processing topology, the relative computation and communication capabilities of the processor array and partitioning the algorithm into tasks and the scheduling of these tasks. It is essential to note that in implementing an algorithm on a parallel computation platform, consideration of

i. the interconnection scheme issues;
ii. the scheduling and mapping of the algorithm on the architecture; and
iii. the mechanism for detecting parallelism and partitioning the algorithm into modules or sub-tasks;

will lead to computational speedup (Agrawal et al., 1986; Crumley et al., 1994).

This chapter presents the real-time processing requirements of the eight different algorithms described in Chapter 6. These were implemented on the heterogeneous and homogeneous parallel processing architectures, described Appendix B, on the basis of computing methods, described in Chapter 6. The uniprocessor architectures investigated, include, an i860 processor, a SUN SPARCstation of TMS390S10 processor type, a T8 transputer processor and a TMS320C40 DSP processor. In contrast, the multiprocessor architectures, include, a homogeneous network of T8s, a heterogeneous architecture of an i860 and a T8, a homogeneous architecture of C40 DSPs and a heterogeneous network comprising a C40 and a T8. All the algorithms are implemented on the four uniprocessor architectures and the four different parallel architectures. The flexible beam and flexible manipulator simulation algorithms were further considered for implementation on more than two processor-based architectures. The 3L Parallel C and ANSI C compilers are used to investigate the performance of the architectures. Finally, comparative real-time computational aspects, including, execution time, speedup and efficiency are looked at to reveal the matching and mismatching of the algorithms with the architectures.

### 8.3 Adaptive Active Vibration Control

As discussed in the previous chapters, the adaptive AVC algorithm consists of three different algorithms, namely, a simulation algorithm, an identification algorithm and a control algorithm. These were implemented on a number of computing platforms. Results of these implementations are presented and discussed with the real-time performance of the algorithms and architectures in this section.