2.1 Basic Model of a real-time system

Fig. 1 shows a simple model of a real-time system in terms of its important functional blocks. Observe that in Fig. 3, the sensors are interfaced with the input conditioning block, which in turns is connected to the input interface. The output interface, output conditioning, and the actuator are interfaced in a complementary manner. In the following we briefly describe the roles of the different functional blocks of a real-time system

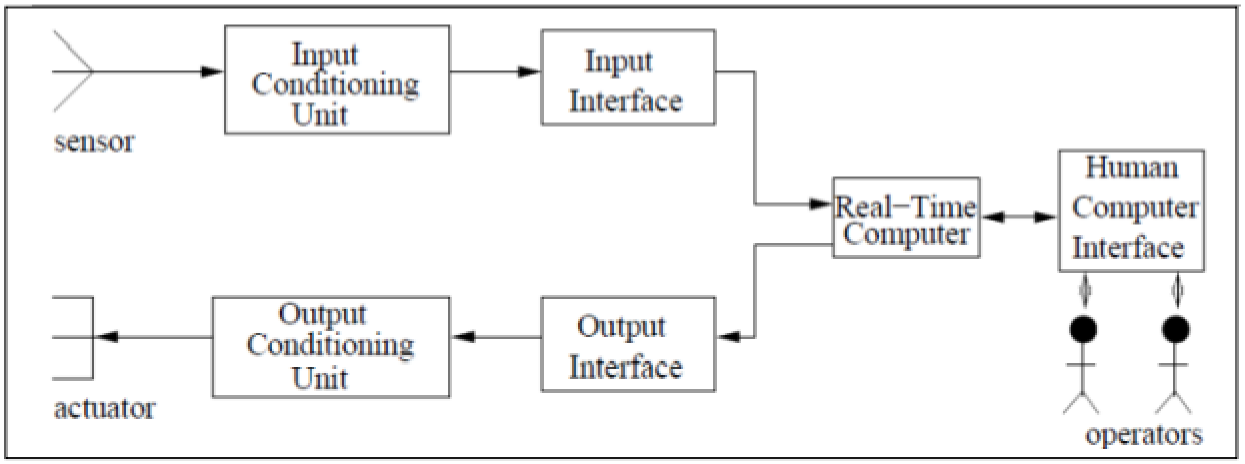

Fig. 1: A Model of a Real-Time System

Sensor: A sensor converts some physical characteristic of its environment into electrical signals. An example of a sensor is photo-voltaic cell which converts light energy into electrical energy. A wide variety of temperature and pressure sensors are also used.

Actuator: an actuator is any device that takes its inputs from the output interface of a computer and converts these electrical signals into some physical actions on its environment. The physical actions may be in the form of motion, change of thermal, electrical, pneumatic, or physical characteristics of some objects. A popular actuator is a motor. Heaters are also very commonly used. Besides, several hydraulic and pneumatic actuators are also popular

Signal Conditioning Units: The electrical signals produced by a computer can rarely be used to directly drive an actuator. The computer signals usually need conditioning before they can be used by the actuator. This is termed output conditioning. Similarly, input conditioning is required to be carried out on sensor signals before they can be accepted by the computer. For example, analog signals generated by a photo-voltaic cell are normally in the mille-volts range and need to be conditioned before they can be processed by a computer. The following are some important types of conditioning carried out on raw signals generated by sensors and digital signal generated by computer:

1- **Voltage amplification:** voltage amplification is normally required to be carried out to match the full scale sensor voltage output with the full scale voltage input to the interface of a computer. For example, a sensor might produce voltage in the mille-volts range; whereas the input interface of a computer may require the input signal level to be of the order of a volt.

2- **voltage level shifting:** voltage level shifting is often required to align the voltage level generated by a sensor with that acceptable to the computer. For example, a sensor may produce voltage in the range -0.5 to +0.5 volt, whereas the input interface of the computer may accept voltage only in the range of 0 to 1 volt. In this case, the sensor voltage must undergo level shifting before it can be used by the computer.

3- **frequency range shifting and filtering**: frequency range shifting is often used to reduce the noise components in a signal. Many types of noise occur in narrow bands and the signal must be shifted from the noise bands so that noise can be filtered out.

4- **Signal mode conversion**: a type of signal mode conversion that is frequently carried out during signal conditioning involves changing direct current into alternating current and vice-versa. Another type signal mode conversion that is frequently used is conversion of analog signals to a constant amplitude pulse train such that the pulse rate or pulse width is proportional to the voltage level. Conversion of analog signals to a pulse train is often necessary for input to systems such as transformer coupled circuits that do not pass direct current.

2.2 Characteristics of Real Time System

We now discuss a few key characteristics of real-time systems. These characteristics distinguish real-time systems from non-real-time systems. However, the reader may note that all the discussed characteristics many not be applicable to every real-time system.

1- Time constraints: every real-time task is associated with some time constraints. One form of time constraints that is very common is deadlines associated with tasks. A task deadline specifies the time before which the task must complete and produce the results. Other type of timing constraints are delay and duration. It is the responsibility of the real-time operating system (RTOS) to ensure that all tasks meet their respective time constraints. Time constraints can be classified into the following three types;

a- Delay constraint: a delay constraint captures the minimum time (delay) that must elapse between the occurrence of two arbitrary events e1 and e2.

b- Deadline constraint: a deadline constraint captures the permissible maximum separation between any two arbitrary events e1 and e2.

c- Duration constraints: a duration constraint on an event specifies the time period over which the event acts. A duration constraint can either be minimum type or maximum type. The minimum type duration constraint requires that once the event starts the event must not end before a certain minimum duration. Whereas a maximum type duration constraint requires that once the event starts, the event must end before a certain maximum duration elapses.

2- New correctness criterion: the notion of correctness in real-time systems is different from that used in the context of traditional systems. In real-time systems, correctness implies not only logical correctness of the results, but the time at which the results are produced is important. A logically correct result produced after the deadline would be considered as an incorrect result.

3- Embedded: A vast majority of real-time systems are embedded in nature. An embedded computer system is physically "embedded" in its environment and often controls it. The sensors of the real-time computer collect data from the environment pass them on to the real-time computer for processing. The computer, in turn passes information (processed data) to the actuators to carry out the necessary work on the environment, which results in controlling some characteristics of the environment. An example of an embedded real-time system is Multi-Point Fuel Injection (MPFI) system.

4- Safety-Criticality: For traditional non-real-time system safety and reliability are independent issues. However, in many real-time systems these two issues are intricately bound together making them safety-critical. Note that a safe system is one that does not cause any damage even when it fails. A reliable system is one that can operate for long durations of time without exhibiting any failures.

5- Concurrency: A real-time system usually needs to respond to several independent events within very short and strict time bounds. For instance, consider a chemical plant automation system, which monitors the progress of a chemical reaction and controls the rate of reaction. These parameters are sensed using sensors fixed in the chemical reaction chamber. These sensors may generate data asynchronously at different rates. Therefore, the real-time system must process data from all the sensors concurrently, otherwise signals may be lost and the system may malfunction.

6- Distributed and Feedback Structure: in many real-time systems, the different components of the system are naturally distributed across widely spread geographic locations. Therefore, these events may often have to be handled locally and responses produced to them to prevent overloading of the underlying communication network.

Therefore, the sensors and the actuators may be located at the places where the events are generated.

7- Task Criticality: task criticality is a measure of the cost of failure of a task. Task criticality is determined by examining how critical are the results produced by the task to the proper functioning of the system. A real-time system may have tasks of very different criticalities. It is therefore natural to expect that the criticalities of the different tasks must be taken into consideration while designing for fault-tolerance. The higher the criticality of a task, the more reliable it should be made. Further, in the event of the failure of a highly critical task, immediate failure detection and recovery are important.

8- Custom Hardware: A real-time system is often implemented on custom hardware that is specifically designed and developed for the purpose. For example, an MPFI car used a processor that must be powerful general purpose processor such as a Pentium or an Athlon processor. Some of the most powerful computers used in MPFI engines are 16- or 32-bit processors running at approximately 40 MHz. However, unlike the conventional PCs, a processor used in these car engines do not deal with processing frills such as screen-savers or a dozens of different applications running at the same time. All that the processor in an MPFI system need to do is to compute the required fuel injection rate that is most efficient for a given speed and acceleration.

9- Reactive: real-time systems are often reactive. A reactive system is one in which an on-going interaction between the computer and the environment is maintained. Traditional systems compute functions on the input data to generate the output data. In contrast to traditional computation of the output as a simple function of the input data, real-time systems do not produce any output data but enter into on-going interaction with their environment. In each interaction step, the results computed are used to carry out some actions on the environment. The reaction of the environment is sampled and is fed back to the system. Therefore, the computations in a real-time system can be considered to be non-terminating.

10- 10.Stability: Under overload conditions, real-time systems need to continue to meet the deadline of the most critical tasks, though the deadlines of non-critical task may not be met. This is in contrast to the requirement of fairness for traditional systems even under overload conditions.

11- 11.Exception Handling: Many real-time systems work round-the-clock and often operate without human operators. For example, consider a small automated chemical plant that is set up to work non-stop. When there are no human operators, taking corrective actions on a failure become difficult. Even if no corrective actions can be immediate taken, it is desirable that a failure does not result in catastrophic situations. A failure should be detected and the system should continue to operate in a gracefully degraded mode rather than shutting off abruptly.