Real-time Systems Design

- Designing embedded software systems whose behavior is subject to timing constraints

Topics covered
- Systems design
- State machine modeling
- Real-time executives
- Monitoring and control systems
- Data acquisition systems
Real-time systems

- Systems which monitor and control their environment
- Inevitably associated with hardware devices
  - Sensors: Collect data from the system environment
  - Actuators: Change (in some way) the system's environment
- Time is critical. Real-time systems MUST respond within specified times.
- A 'soft' real-time system is a system whose operation is DEGRADED if results are not produced according to the specified timing requirements.
- A 'hard' real-time system is a system whose operation is INCORRECT if results are not produced according to the timing specification.

Stimulus/Response Systems

- Stimulus = input event
- Given a stimulus, the system must produce a response within a specified time
- Periodic stimuli. Stimuli which occur at predictable time intervals
  - For example, a temperature sensor may be polled 10 times per second
- Aperiodic stimuli. Stimuli which occur at unpredictable times
  - For example, a system power failure may trigger an interrupt which must be processed by the system

Architectural considerations

- Because of the need to respond to timing demands made by different stimuli/responses, the system architecture must allow for fast switching between stimulus handlers.
- Timing demands of different stimuli are different so a simple sequential loop is not usually adequate.
- Real-time systems are usually designed as cooperating processes with a real-time executive controlling these processes.
A real-time system model

System elements:
- Sensors control processes
- Data processor
- Actuator control

Real-time control system

Actuator
Actuator
Actuator
Actuator

Sensor
Sensor
Sensor
Sensor
Sensor
Sensor

Sensor/actuator processes

Sensor
Actuator

Stimulus
Response

Sensor control
Data processor
Actuator control

The three processes execute concurrently

System design

- Design decisions should be made on the basis on non-functional system requirements
- Design both the hardware and the software associated with the system.
- Hardware delivers better performance but potentially longer development and less scope for change
Real-time systems design process

- Identify the stimuli to be processed and the required responses to these stimuli
- For each stimulus and response, identify the timing constraints
- Aggregate the stimulus and response processing into concurrent processes.
- A process may be associated with each class of stimulus and response.
- Design algorithms to process each class of stimulus and response. These must meet the given timing requirements.
- Design a scheduling system which will ensure that processes are started in time to meet their deadlines.
- Integrate using a real-time executive or operating system.

Timing constraints

- May require extensive simulation and experiment to ensure that these are met by the system.
- May mean that certain design strategies such as object-oriented design cannot be used because of the additional overhead involved.
- May mean that low-level programming language features have to be used for performance reasons.

State machine modeling

- The effect of a stimulus in a real-time system may trigger a transition from one state to another.
- Finite state machines can be used for modeling real-time systems.
- Process
  - make FSM diagrams
  - list and describe the states
  - list and describe the stimuli
Microwave oven state machine

Microwave oven states

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Half power on</td>
<td>The oven power is set to 300 watts</td>
</tr>
<tr>
<td>Full power on</td>
<td>The oven power is set to 600 watts</td>
</tr>
<tr>
<td>Set time</td>
<td>The cooking time is set to the user’s input value</td>
</tr>
<tr>
<td>Operation disabled</td>
<td>Oven operation is disabled for safety. Interior oven light is on</td>
</tr>
<tr>
<td>Operation enabled</td>
<td>Oven operation is enabled. Interior oven light is off</td>
</tr>
<tr>
<td>Timed operation</td>
<td>Oven in operation. Interior oven light is on. Timer is counting down.</td>
</tr>
<tr>
<td>Cooking complete</td>
<td>Timer has counted down to zero. Audible alarm signal is on. Oven light is off</td>
</tr>
</tbody>
</table>

Microwave oven stimuli

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Half power</td>
<td>The user has pressed the half power button</td>
</tr>
<tr>
<td>Full power</td>
<td>The user has pressed the full power button</td>
</tr>
<tr>
<td>Timer</td>
<td>The user has pressed one of the timer buttons</td>
</tr>
<tr>
<td>Door open</td>
<td>The oven door switch is not closed</td>
</tr>
<tr>
<td>Door closed</td>
<td>The oven door switch is closed</td>
</tr>
<tr>
<td>Start</td>
<td>The user has pressed the start button</td>
</tr>
<tr>
<td>Timeout</td>
<td>Timer signal indicating that set cooking time is finished</td>
</tr>
</tbody>
</table>

We reserve all rights in this document and in the information contained therein. Reproduction, use or disclosure to third parties without express authority is strictly forbidden. Mdh/Idt, Ivica Crnkovic ©
Thread diagrams

- FSM models lack structure. Even simple systems can have a complex model.
- Thread diagrams which show an event sequence are a means of managing the complexity in state machine models.
- Used to structure and present state model information
- Show the end-to-end processing of a specific set of stimuli
- Thread diagrams should be produced for message combinations. This means for systems with multiple stimuli, this approach may be impractical

Thread diagram - full power

Real-time executives

- Real-time executives are specialized operating systems which manage the processes in the RTS
- Responsible for process management and resource (processor and memory) allocation
- May be based on a standard RTE kernel which is used unchanged or modified for a particular application
- Usually does not include facilities such as file management
Executive components

- Real-time clock
  - Provides information for process scheduling.
- Interrupt handler
  - Manages aperiodic requests for service.
- Scheduler
  - Chooses the next process to be run.
- Resource manager
  - Allocates memory and processor resources.
- Dispatcher
  - Starts process execution.

Real-time executive components

Process priority

- The processing of some types of stimuli must sometimes take priority
- Interrupt level priority. Highest priority which is allocated to processes requiring a very fast response
- Clock level priority. Allocated to periodic processes
- Within these, further levels of priority may be assigned
Interrupt servicing

- Control is transferred automatically to a pre-determined memory location
- This location contains an instruction to jump to an interrupt service routine
- Further interrupts are disabled, the interrupt serviced and control returned to the interrupted process
- Interrupt service routines MUST be short, simple and fast

Periodic process servicing

- In most real-time systems, there will be several classes of periodic process, each with different periods (the time between executions), execution times and deadlines (the time by which processing must be completed)
- The real-time clock ticks periodically and each tick causes an interrupt which schedules the process manager for periodic processes
- The process manager selects a process which is ready for execution

Process switching

- The scheduler chooses the next process to be executed by the processor. This depends on a scheduling strategy which may take the process priority into account
- The resource manager allocates memory and a processor for the process to be executed
- Dispatcher takes process from ready list, loads it onto a processor and starts execution
Some examples of R-T systems

- Monitoring and Control systems
  - Check sensors providing information and take actions depending on the sensor’s state
  - Monitoring systems examine sensors and report their results
  - Control systems collect data from sensors but no real-time actuator control

- Data acquisition systems
  - Collect data from sensors for subsequent processing and analysis.

Data acquisition systems

- Data collection processes and processing processes may have different periods and deadlines.
- Data collection may be faster than processing e.g. collecting information about an explosion.
- Circular or ring buffers are a mechanism for smoothing speed differences.

Example: Reactor data collection

- System collects data from a set of sensors monitoring the neutron flux from a nuclear reactor.
- Flux data is placed in a ring buffer for later processing.
- The ring buffer is itself implemented as a concurrent process so that the collection and processing processes may be synchronized.
Mutual exclusion

- Producer processes collect data and add it to the buffer. Consumer processes take data from the buffer and make elements available.
- Producer and consumer processes must be mutually excluded from accessing the same element.
- The buffer must stop producer processes adding information to a full buffer and consumer processes trying to take information from an empty buffer.

Key points

- Real-time system correctness depends not just on what the system does but also on how fast it reacts.
- Delay partitioning of functions to hardware and software until as late as possible in the design process.
- Real-time systems are usually designed as a number of concurrent processes.
- A R-T system model may associate processes with each class of sensor and actuator.
- Real-time executives are responsible for process and resource management.