Real-Time Software Workshop

Workshop Solutions “Light Controller”

Onno van Roosmalen
O.S.vanRoosmalen@chello.nl
Composite Structure Design

- Rc5 : RC5
- control
- rc
- lightLevel
- remainingTime
- newTimeOut
- ld: LM
- lD: PD
- pd: PD
- :Timer
- :Dali
- daliHW
- :I2C
- nvRam
- memory
- lightSensor
- presenceDetector
- rcReceiver
## Response Times

<table>
<thead>
<tr>
<th>Process</th>
<th>Response Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dali</td>
<td>100 $\mu$s</td>
</tr>
<tr>
<td>RC5</td>
<td>170 $\mu$s</td>
</tr>
<tr>
<td>Timer</td>
<td>200 $\mu$s</td>
</tr>
<tr>
<td>LM</td>
<td>$\mu$s</td>
</tr>
<tr>
<td>I2C</td>
<td>$\mu$s</td>
</tr>
<tr>
<td>PD</td>
<td>$\mu$s</td>
</tr>
<tr>
<td>Main</td>
<td>$\mu$s</td>
</tr>
</tbody>
</table>

*Response not within deadline*

100 $\mu$s > 41 $\mu$s

*No valid response time within period*
Timing Diagram

Dali
RC5
Timer
LM
I2C
PD
Meeting Timing Requirements

• The Dali response time (100 $\mu$s) is not within the deadline (41 $\mu$s).
• One could try to put the daliHW instructions early in the body of the Dali process body.
• However, doing so it is still difficult to analyze if the daliHW settings will be timely.
• Also, this will not work if there is more than one dali to control
Dali Process

- Proper solution: split the Dali process into two parts

```
Dali [dali] -> DaliOut [daliHW]
    |                         |
    v                         
memory [memory] -> I2C [I2C] -> element [element]
```
Dali Process

\[ P_{\text{Dali}}(T) ::= \]
\[
\text{state} = \text{Idle};
\]
\[
\text{while}\text{(waitForNextPeriod}(T)) \{ \\
\quad \text{if}(\text{state} == \text{Idle}) \{ \\
\quad\quad \text{if}(\text{!dali.isEmpty}()) \{ \\
\quad\quad\quad \text{element} = \text{start}; \\
\quad\quad\quad \text{daliCommand} = \text{dali.get}(); \\
\quad\quad\quad \text{memory.put}(\text{daliCommand}, \text{address}); \\
\quad\quad\quad \text{state} = \text{Sending}; \\
\quad\quad \} \\
\quad \} \text{else if (state == Sending) } \{ \\
\quad \text{if}(\text{daliCommand.atEnd}()) \{ \\
\quad\quad \text{element} = \text{stop}; \\
\quad\quad \text{state} = \text{Idle}; \\
\quad \} \text{ else } \{ \\
\quad\quad \text{element} = \text{daliCommand.nextElement}(); \\
\quad \} \\
\} \\
\} \]
DaliOut Process

\[ P_{\text{DaliOut}}(T) ::= \]
\[
\text{while(waitForNextPeriod}(T)) \{ \\
\text{daliHW}=\text{element}; \\
\}
\]
Dali: Remark

• 25 ms between dali commands
• The dali should also use a timer.
  – This timer can be merged with the one we already have (temporal cohesion).
## Timings

<table>
<thead>
<tr>
<th>Module</th>
<th>Ci (μs)</th>
<th>Ti (μs)</th>
<th>Di (μs)</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>DaliOut</td>
<td>10</td>
<td>416</td>
<td>41</td>
<td>1</td>
</tr>
<tr>
<td>RC5</td>
<td>70</td>
<td>222</td>
<td>=Ti</td>
<td>2</td>
</tr>
<tr>
<td>Timer</td>
<td>30</td>
<td>250</td>
<td>=Ti</td>
<td>3</td>
</tr>
<tr>
<td>LM</td>
<td>30</td>
<td>250</td>
<td>=Ti</td>
<td>4</td>
</tr>
<tr>
<td>Dali</td>
<td>100</td>
<td>416</td>
<td>=Ti</td>
<td>5</td>
</tr>
<tr>
<td>I2C</td>
<td>40</td>
<td>500</td>
<td>=Ti</td>
<td>6</td>
</tr>
<tr>
<td>PD</td>
<td>30</td>
<td>10000</td>
<td>=Ti</td>
<td>7</td>
</tr>
<tr>
<td>Main</td>
<td>400</td>
<td>40000</td>
<td>=Ti</td>
<td>8</td>
</tr>
</tbody>
</table>
Assignment 9

• To proof schedulability we must compute the response times of the various tasks in the modified design.
## Response Times

<table>
<thead>
<tr>
<th>Process</th>
<th>Response Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DaliOut</td>
<td>10 µs</td>
</tr>
<tr>
<td>RC5</td>
<td>80 µs</td>
</tr>
<tr>
<td>Timer</td>
<td>110 µs</td>
</tr>
<tr>
<td>LM</td>
<td>140 µs</td>
</tr>
<tr>
<td>Dali</td>
<td>370 µs</td>
</tr>
<tr>
<td>I2C</td>
<td>410 µs</td>
</tr>
<tr>
<td>PD</td>
<td>1200 µs</td>
</tr>
<tr>
<td>Main</td>
<td>4960 µs</td>
</tr>
</tbody>
</table>

*Response within deadline 41 µs*