Texas Instruments Robotics System Learning Kit
Module 13

Lecture: Timers – Periodic Interrupts
Periodic interrupts using Timers

You will learn in this module

- Timer A on the TI MSP432
  - Clock input, prescale
  - Counter

- Periodic interrupts
  - Additional threads
  - Priority
Timer - A for input and output

Timer A Features

- Pins
  - Input capture
  - Output compare

- Precision
  - 16-bits

- Resolution
  - Clock period
  - Prescale

\[ SMCLK = \frac{48\text{MHz}}{4} = 12\text{ MHz}, 83.33\text{ns} \]
## Timer - A registers

<table>
<thead>
<tr>
<th>Name</th>
<th>15-10</th>
<th>9-8</th>
<th>7-6</th>
<th>5-4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASSEL ID</td>
<td>MC</td>
<td>TA0CTL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0EX0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0IV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA0CCR6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### timers and periodic interrupts

- **Timers**
  - 16-bit counter
  - 16-bit Capture/Compare 0 Register
  - 16-bit Capture/Compare 1 Register
  - 16-bit Capture/Compare 2 Register
  - 16-bit Capture/Compare 3 Register
  - 16-bit Capture/Compare 4 Register
  - 16-bit Capture/Compare 5 Register
  - 16-bit Capture/Compare 6 Register

- **Periodic Interrupts**
  - TA0CTL
  - TA0EX0
  - TA0IV
Clock and prescale

<table>
<thead>
<tr>
<th>TASSEL</th>
<th>Selected Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>TAxCLK</td>
</tr>
<tr>
<td>01</td>
<td>ACLK</td>
</tr>
<tr>
<td>10</td>
<td>SMCLK</td>
</tr>
<tr>
<td>11</td>
<td>INCLK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Prescale</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>/1</td>
</tr>
<tr>
<td>01</td>
<td>/2</td>
</tr>
<tr>
<td>10</td>
<td>/4</td>
</tr>
<tr>
<td>11</td>
<td>/8</td>
</tr>
</tbody>
</table>

Resolution = $T \times 2^{ID} \times (TADEX+1)$
Range = Precision \times Resolution
## Timer - A Modes

<table>
<thead>
<tr>
<th>MC</th>
<th>Mode control</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Stop</td>
</tr>
<tr>
<td>01</td>
<td>Up mode: Timer counts up to TAxCCR0</td>
</tr>
<tr>
<td>10</td>
<td>Continuous mode: Timer counts up to 0xFFFF</td>
</tr>
<tr>
<td>11</td>
<td>Up/down mode: Timer counts up to TAxCCR0 then down to 0x0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OUTMOD</th>
<th>On match to TAxCCRY</th>
<th>On match to TAxCCR0</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>OUT bit value</td>
<td></td>
</tr>
<tr>
<td>001</td>
<td>Set</td>
<td></td>
</tr>
<tr>
<td>010</td>
<td>Toggle</td>
<td>Reset</td>
</tr>
<tr>
<td>011</td>
<td>Set</td>
<td>Reset</td>
</tr>
<tr>
<td>100</td>
<td>Toggle</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>Reset</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>Toggle</td>
<td>Set</td>
</tr>
<tr>
<td>111</td>
<td>Reset</td>
<td>Set</td>
</tr>
</tbody>
</table>
Clock and prescale

0) Halt the timer (MC=00),
1) Set the timer clock and prescale,
2) Set submodule 0 to compare, arm interrupt
3) Set the TAxCCR0 to the interrupt period minus 1
4) Set the priority in the correct NVIC Priority register
5) Enable the interrupt in the NVIC Interrupt Enable register
6) Reset the timer and start it in up mode
7) Enable interrupts (in the main program after all devices initialized)
## Interrupt Vectors, numbers, names, and priority

<table>
<thead>
<tr>
<th>Vector</th>
<th>Number</th>
<th>IRQ</th>
<th>ISR name</th>
<th>NVIC priority</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000002C</td>
<td>11</td>
<td>-5</td>
<td>SVC_Handler</td>
<td>SCB_SHPR2</td>
<td>31 – 29</td>
</tr>
<tr>
<td>0x00000038</td>
<td>14</td>
<td>-2</td>
<td>PendSV_Handler</td>
<td>SCB_SHPR3</td>
<td>23 – 21</td>
</tr>
<tr>
<td>0x0000003C</td>
<td>15</td>
<td>-1</td>
<td>SysTick_Handler</td>
<td>SCB_SHPR3</td>
<td>31 – 29</td>
</tr>
<tr>
<td>0x00000060</td>
<td>24</td>
<td>8</td>
<td>TA0_0_IRQHandler</td>
<td>NVIC_IPR2</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x00000064</td>
<td>25</td>
<td>9</td>
<td>TA0_N_IRQHandler</td>
<td>NVIC_IPR2</td>
<td>15 – 13</td>
</tr>
<tr>
<td>0x00000068</td>
<td>26</td>
<td>10</td>
<td>TA1_0_IRQHandler</td>
<td>NVIC_IPR2</td>
<td>23 – 21</td>
</tr>
<tr>
<td>0x0000006C</td>
<td>27</td>
<td>11</td>
<td>TA1_N_IRQHandler</td>
<td>NVIC_IPR2</td>
<td>31 – 29</td>
</tr>
<tr>
<td>0x00000070</td>
<td>28</td>
<td>12</td>
<td>TA2_0_IRQHandler</td>
<td>NVIC_IPR3</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x00000074</td>
<td>29</td>
<td>13</td>
<td>TA2_N_IRQHandler</td>
<td>NVIC_IPR3</td>
<td>15 – 13</td>
</tr>
<tr>
<td>0x00000078</td>
<td>30</td>
<td>14</td>
<td>TA3_0_IRQHandler</td>
<td>NVIC_IPR3</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x0000007C</td>
<td>31</td>
<td>15</td>
<td>TA3_N_IRQHandler</td>
<td>NVIC_IPR3</td>
<td>15 – 13</td>
</tr>
<tr>
<td>0x00000080</td>
<td>32</td>
<td>16</td>
<td>EUSCIA0_IRQHandler</td>
<td>NVIC_IPR4</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x00000084</td>
<td>33</td>
<td>17</td>
<td>EUSCIA1_IRQHandler</td>
<td>NVIC_IPR4</td>
<td>23 – 21</td>
</tr>
<tr>
<td>0x00000088</td>
<td>34</td>
<td>18</td>
<td>EUSCIA2_IRQHandler</td>
<td>NVIC_IPR4</td>
<td>31 – 29</td>
</tr>
<tr>
<td>0x0000008C</td>
<td>35</td>
<td>19</td>
<td>EUSCIA3_IRQHandler</td>
<td>NVIC_IPR4</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x00000090</td>
<td>36</td>
<td>20</td>
<td>EUSCIB0_IRQHandler</td>
<td>NVIC_IPR5</td>
<td>15 – 13</td>
</tr>
<tr>
<td>0x00000094</td>
<td>37</td>
<td>21</td>
<td>EUSCIB1_IRQHandler</td>
<td>NVIC_IPR5</td>
<td>23 – 21</td>
</tr>
<tr>
<td>0x00000098</td>
<td>38</td>
<td>22</td>
<td>EUSCIB2_IRQHandler</td>
<td>NVIC_IPR5</td>
<td>31 – 29</td>
</tr>
<tr>
<td>0x0000009C</td>
<td>39</td>
<td>23</td>
<td>EUSCIB3_IRQHandler</td>
<td>NVIC_IPR5</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x000000CC</td>
<td>51</td>
<td>35</td>
<td>PORT1_IRQHandler</td>
<td>NVIC_IPR8</td>
<td>15 – 13</td>
</tr>
<tr>
<td>0x000000DD</td>
<td>52</td>
<td>36</td>
<td>PORT2_IRQHandler</td>
<td>NVIC_IPR9</td>
<td>7 – 5</td>
</tr>
<tr>
<td>0x000000D4</td>
<td>53</td>
<td>37</td>
<td>PORT3_IRQHandler</td>
<td>NVIC_IPR9</td>
<td>15 – 13</td>
</tr>
<tr>
<td>0x000000DB</td>
<td>54</td>
<td>38</td>
<td>PORT4_IRQHandler</td>
<td>NVIC_IPR9</td>
<td>23 – 21</td>
</tr>
<tr>
<td>0x000000DC</td>
<td>55</td>
<td>39</td>
<td>PORT5_IRQHandler</td>
<td>NVIC_IPR9</td>
<td>31 – 29</td>
</tr>
<tr>
<td>0x000000E0</td>
<td>56</td>
<td>40</td>
<td>PORT6_IRQHandler</td>
<td>NVIC_IPR10</td>
<td>7 – 5</td>
</tr>
</tbody>
</table>

```c
void TA2_0_IRQHandler(void){
    TIMER_A2->CCTL[0] &= ~0x0001; // ack
    // body
}
```

Look for `interruptVectors[]` in the file `startup_msp432p401r_ccs.c`
Interrupts enabled in the main program after all devices initialized

\[
T \ast 2^{16} \ast (TAIDEX+1)(TAXCCR0+1) = 2\mu s \ast \text{period}
\]

void (*TimerA2Task)(void); // user function
void TimerA2_Init(void(*task)(void), uint16_t period){
    TimerA2Task = task; // user function
    // bits9-8=10, clock source to SMCLK
    // bits7-6=10, input clock divider /4
    // bits5-4=00, stop mode
    // bit2=0, set this bit to clear
    // bit1=0, no interrupt on timer
    TIMER_A2->CTL = 0x0280;
    // bits15-14=00, no capture mode
    // bit8=0, compare mode
    // bit4=1, enable capture/compare interrupt on CCIFG
    // bit2=0, output this value in output mode 0
    // bit0=0, clear capture/compare interrupt pending
    TIMER_A2->CCTL[0] = 0x0010;
    TIMER_A2->CCR[0] = (period - 1); // compare match value
    TIMER_A2->EX0 = 0x0005; // configure for input clock divider /6
    NVIC->IP[3] = (NVIC->IP[3] & 0xFFFFFFFF) | 0x00000040; // priority 2
    NVIC->ISER[0] = 0x00001000; // enable interrupt 12 in NVIC
    TIMER_A2->CTL |= 0x0014; // reset and start Timer A in up mode
}

void TA2_0_IRQHandler(void){
    TIMER_A2->CCTL[0] &= ~0x0001
    *TimerA2Task();
}

NVIC->IP[2] = (NVIC->IP[2] & 0xFFFF0000) | 0x00400000; // enable interrupt 10 in NVIC
NVIC->ISER[0] = 0x00000040; // enable interrupt 10 in NVIC

Periodic Interrupt

Timers – Periodic Interrupts
Summary

Timer overview
- Clock input
- Prescale
- Counter

Periodic interrupts
- Additional threads
- Priority

```c
void TA0_0_IRQHandler(void)
{
    TIMER_A0->CCTL[0] &= ~0x0001;
    // ack
    // body
}

void TA1_0_IRQHandler(void)
{
    TIMER_A1->CCTL[0] &= ~0x0001;
    // ack
    // body
}

void TA2_0_IRQHandler(void)
{
    TIMER_A2->CCTL[0] &= ~0x0001;
    // ack
    // body
}
```
Module 13

Lecture: Timers – Pulse Width Modulation
PWM using Timers

You will learn in this module

- Timer A on the TI MSP432
  - Clock input, prescale
  - Counter

- PWM output
  - Adjust power to motors
  - Two independent outputs
Interface circuit using TI-RSLK chassis board

PWM on P2.6, P2.7
Using Timer - A for Two PWM Outputs

You will convert it to
TA0.4/P2.7
TA0.3/P2.6
Using Timer - A for Two PWM Outputs

PWM Mode
- Up/down count
  - Up to CCR0 (10)
  - Down to 0
- P2.4=1 when timer equals TA0CCR1 (7) on way down
- P2.4=0 when timer equals TA0CCR1 (7) on way up

\[
\text{Dutycycle} = \frac{\text{High}}{\text{High} + \text{Low}} = \frac{\text{High}}{\text{Period}}
\]

\[
T = \text{Prescale}/12\text{MHz}
\]

\[
\text{Period} = 2 \times T \times CCR0
\]

\[
\text{DutyCycle} = \frac{\text{CCR1}}{\text{CCR0}}
\]
PWM Outputs on P2.4 and P2.5

// SMCLK = 48MHz/4 = 12 MHz, 83.33ns
// Counter counts up to TA0CCR0 and back down
// Let Timer clock period T = 8/12MHz = 666.7ns
// Period of P2.4 is period*1.333us, duty cycle is duty1/period
// Period of P2.5 is period*1.333us, duty cycle is duty2/period

void PWM_Init12(uint16_t period, uint16_t duty1, uint16_t duty2){
    P2->DIR |= 0x30;          // P2.4, P2.5 output
    P2->SEL0 |= 0x30;         // P2.4, P2.5 Timer0A functions
    P2->SEL1 &= ~0x30;        // P2.4, P2.5 Timer0A functions
    TIMER_A0->CCTL[0] = 0x0080; // CCI0 toggle
    TIMER_A0->CCR[0] = period;       // Period is 2*period*8*83.33ns is 1.333*period
    TIMER_A0->EX0 = 0x0000;         // divide by 1
    TIMER_A0->CCTL[1] = 0x0040;     // CCR1 toggle/reset
    TIMER_A0->CCR[1] = duty1;       // CCR1 duty cycle is duty1/period
    TIMER_A0->CCTL[2] = 0x0040;     // CCR2 toggle/reset
    TIMER_A0->CCR[2] = duty2;       // CCR2 duty cycle is duty2/period
    TIMER_A0->CTL = 0x02F0;        // SMCLK=12MHz, divide by 8, up-down mode

    // bit mode
    // 9-8 10  TASSEL, SMCLK=12MHz
    // 7-6 11  ID, divide by 8
    // 5-4 11  MC, up-down mode
    // 2  0  TACLR, no clear
    // 1  0  TAIE, no interrupt
    // 0          TAIFG
}

T = 8/12MHz
Period = 2*T*15000 = 10ms
DutyCycle = CCR1/15000

You will convert it to TA0.4/P2.7
TA0.3/P2.6
PWM Outputs on P2.4 and P2.5

```c
//****************************************PWM_Duty1*************************************************
// change duty cycle of PWM output on P2.4
// Inputs: duty1
// Outputs: none
// period of P2.4 is 2*period*666.7ns, duty cycle is duty1/period
void PWM_Duty1(uint16_t duty1){
    TIMER_A0->CCR[1] = duty1; // CCR1 duty cycle is duty1/period
}

//****************************************PWM_Duty2*************************************************
// change duty cycle of PWM output on P2.5
// Inputs: duty2
// Outputs: none// period of P2.5 is 2*period*666.7ns, duty cycle is duty2/period
void PWM_Duty2(uint16_t duty2){
    TIMER_A0->CCR[2] = duty2; // CCR2 duty cycle is duty2/period
}
```

**DutyCycle**

- Precision 14999 alternatives (0 to 14998)
- Range 0 to 99.99%
- Resolution 0.0067%
Summary

PWM output
- Adjust power to motors
- Two independent outputs

![Diagram showing PWM output and timer functionality](image)

- **PWM output**
  - Adjust power to motors
  - Two independent outputs
IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2019, Texas Instruments Incorporated