CmpE 443 - PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

Problem Session: Analog to Digital Converter (ATD)
What is ATD?

- Analog to digital converter has a very critical role in embedded systems.
- In real-world, signals are analog.
- Our microcontroller has 8-channel, 8-bit or 10-bit ATD.
- Mainly 4 control registers are used for configuration; ATDCTL2-5
- There are 8 16-bit result registers; ATDDDR0-7
- While higher bytes of result registers are ATDDDRxH, lower bytes are ATDDDRxL
ATD Power up (ATDCTL2)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADPU</td>
<td>AFFC</td>
<td>AWAI</td>
<td>ETRIGLE</td>
<td>ETRIGP</td>
<td>ETRIGE</td>
<td>ASCIE</td>
<td>ASCIF</td>
</tr>
</tbody>
</table>

- **ADPU**: Power up
  - 0: power down
  - 1: normal A/D function
- **AFFC**: Fast Flag Clear
  - 0: A/D conversion complete flag clearing operates normally
  - 1: Enable fast flag clear operation
ATD Conversion Sequence (ATDCTL3)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>S8C</td>
<td>S4C</td>
<td>S2C</td>
<td>S1C</td>
<td>FIFO</td>
<td>FRZ1</td>
<td>FRZ0</td>
</tr>
</tbody>
</table>

- **S8C:S1C : Conversion Sequence Length**
  - Default is 4
  - Exceptions are 0000 and 1xxx which mean 8
- **FIFO: Result Register First in First out Mode**
ATD Resolution (ATDCTL4)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRES8</td>
<td>SMP1</td>
<td>SMP0</td>
<td>PRS4</td>
<td>PRS3</td>
<td>PRS2</td>
<td>PRS1</td>
<td>PRS0</td>
</tr>
</tbody>
</table>

- **SRES8**: Resolution Select
  - 0: 10-bit resolution
  - 1: 8-bit resolution
ATD Data Justification (ATDCTL5)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>DJM</td>
<td>DSGN</td>
<td>SCAN</td>
<td>MULT</td>
<td>0</td>
<td>CC</td>
<td>CB</td>
<td>CA</td>
</tr>
</tbody>
</table>

- DJM: Result Register Data Justification
  - 0: Left justified
  - 1: Right justified
- DSGN: Result Register Data Signed or Unsigned
  - 0: Unsigned data
  - 1: Signed data
- CC, CB, CA: Analog Channel Select Code
ATD Polling (ATDSTAT0)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCF</td>
<td>0</td>
<td>ETORF</td>
<td>FIFOR</td>
<td>0</td>
<td>CC2</td>
<td>CC1</td>
<td>CC0</td>
</tr>
</tbody>
</table>

- **SCF**: Sequence Complete Flag
  - 0: Conversion sequence has not been completed
  - 1: Conversion sequence completed
void main(void) {
    char i;
    unsigned int atd_val;
    ATDCTL2 = 0x80;
    for(i = 0; i<20; i++) ;
    ATDCTL3 = 0x20;
    ATDCTL4 = 0xE3;
    for(;;) {
        ATDCTL5 = 0x81;
        while(ATDSTAT0_SCF == 0) ;
        atd_val = (ATDDRL0 + ATDDRL1 + ATDDRL2 + ATDDRL3)/4;
    }
}
ATD Interrupt (ATDCTL2)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADPU</td>
<td>AFFC</td>
<td>AWAI</td>
<td>ETRIGLE</td>
<td>ETRIGP</td>
<td>ETRIGE</td>
<td>ASCIE</td>
<td>ASCIF</td>
</tr>
</tbody>
</table>

- **ASCIE**: Sequence Complete Interrupt Enable
  - 0: Disabled
  - 1: Enabled

- **ASCIF**: Sequence Complete Interrupt Flag
  - If ASCIE=1, the same as SCF