Universal Asynchronous Receiver / Transmitter (UART)

Last updated 6/17/19
UART

- MSP432 UART
UART

• MSP432 UART
  • ARM (AMBA Compliant)
  • Asynchronous operation
  • 7/8 bit transmission
  • Master/Slave
  • LSB/MSB first
  • Separate RX/TX registers

• 4 eUSCI_A modules
UART

- UART/USART
  - Serial receiver / transmitter
  - Asynchronous and Synchronous versions
  - Asynchronous
    - 2 pin interface
    - RxD – receive data pin
    - TxD – transmit data pin
    - Clock recovery system
  - Synchronous
    - 3 pin interface
    - RxD – receive data pin
    - TxD – transmit data pin
    - Xck – Clock – PD4
UART

- Frame
  - 1 start bit
  - 5, 6, 7, 8, or 9 data bits – typically LSB first
  - none, even, or odd - parity bit
  - 1 or 2 stop bits,
  - Overflow, Framing, Parity – error detection

Bits inside brackets are optional

<table>
<thead>
<tr>
<th>St</th>
<th>Start bit, always low.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n)</td>
<td>Data bits (0 to 8).</td>
</tr>
<tr>
<td>P</td>
<td>Parity bit. Can be odd or even.</td>
</tr>
<tr>
<td>Sp</td>
<td>Stop bit, always high.</td>
</tr>
<tr>
<td>IDLE</td>
<td>No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.</td>
</tr>
</tbody>
</table>
USART

• Synchronous mode operation
  • Master creates clock signal
    • $Xck$ max is typically $\frac{Clk_{system}}{4}$ for timing purposes
  
• Master and Slave
  • Transmit on one clock edge
  • Receive (latch data) on the opposite clock edge
UART

- Asynchronous mode operation
  - Transmit is unchanged
  - Xck is disabled
    - No Master or Slave
  - Must establish an agreed BAUD rate
    - Max BAUD rate is typically $\frac{\text{Clk}_{\text{system}}}{16}$
    - Limited by HW – clock selection options
    - Fixed in SW
    - Start at known BAUD rate and agree to go faster/slower

- Receiver circuitry includes:
  - Clock recovery block
  - Data recovery block
UART

- Asynchronous mode operation

- Clock and Data recovery
  - Internal clock at 16x BAUD rate
  - Sample RxD signal with internal clock
  - Detect Start bit falling edge
  - Sample RxD after 8,9,10 internal clocks
  - Majority determines bit value
  - Assuming a valid start – all further bits sampled at multiples of 16 clocks
UART

- Multi-processor mode
  - Multiple processors sharing the same USART signals
  - Use the last bit in the data (e.g. bit 9 when using 8 bit data) to indicate an address or data value is in the frame
  - If it is an address and it is your address, collect subsequent data frames
  - If it is not your address, ignore subsequent data frames
UART

• Error detection

• Parity
  • Create an error if parity is wrong

• Overflow
  • Create an error if new data is ready to be sampled and the last data has not been read from the buffer yet
  • Double buffering is common

• Frame error
  • Stop bit not detected when expected
  • Idle not detected when expected
## UART

- Common BAUD rates

<table>
<thead>
<tr>
<th>BAUD Rate</th>
<th>Bit Width (us)</th>
<th>Frame Length (us)</th>
<th>Data Rate (Bits/s)</th>
<th>Data Rate (Bytes/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4800</td>
<td>208.33</td>
<td>2291.67</td>
<td>3,491</td>
<td>436</td>
</tr>
<tr>
<td>9600</td>
<td>104.17</td>
<td>1145.83</td>
<td>6,982</td>
<td>873</td>
</tr>
<tr>
<td>19200</td>
<td>52.08</td>
<td>572.92</td>
<td>13,964</td>
<td>1,745</td>
</tr>
<tr>
<td>38400</td>
<td>26.04</td>
<td>286.46</td>
<td>27,927</td>
<td>3,491</td>
</tr>
<tr>
<td>57600</td>
<td>17.36</td>
<td>190.97</td>
<td>41,891</td>
<td>5,236</td>
</tr>
<tr>
<td>115200</td>
<td>8.68</td>
<td>95.49</td>
<td>83,782</td>
<td>10,473</td>
</tr>
</tbody>
</table>
UART

• MSP432 UART Implementation
  • Clock Selector
  • Prescaler / baud rate generator
  • RX/TX buffers
  • RX//TX state machines
  • Error detector
  • Automatic baud rate detection
UART

• MSP432 UART Implementation
  • Required bits
    • start
    • Data 0-7
    • Stop
  • Optional bits
    • data bit 8
    • address bit (7 bit mode)
    • parity bit
    • 2\textsuperscript{nd} stop bit

![Figure 22-2. Character Format](image.png)
• MSP432 UART Implementation
  • Idle Line Microprocessor Format
    • Used when more than 2 devices share the signal paths
    • Blocks of data separated by at least 10 sequential 1's after a stop
    • The first block of data following an idle time is an address
• MSP432 UART Implementation
  • Address Bit Microprocessor Format
    • Used when more than 2 devices share the signal paths
    • The first block of data after a start contains an extra bit to indicate it is an address

*Figure 22-4. Address-Bit Multiprocessor Format*
UART

• MSP432 UART Implementation
  • Automatic baud rate detection
    • Break -> delimiter -> sync pulses
    • Baud rate is calculated based on the sync pulses
      • Max = 1MBaud
      • Min – normal mode = 244 Baud
      • Min – low frequency mode = 15Baud

Figure 22-5. Auto Baud-Rate Detection – Break/Synch Sequence
UART

- MSP432 UART Implementation
  - Receive Error Detection

### Table 22-1. Receive Error Conditions

<table>
<thead>
<tr>
<th>Error Condition</th>
<th>Error Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framing error</td>
<td>UCFE</td>
<td>A framing error occurs when a low stop bit is detected. When two stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>are used, both stop bits are checked for framing error. When a framing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>error is detected, the UCFE bit is set.</td>
</tr>
<tr>
<td>Parity error</td>
<td>UCPE</td>
<td>A parity error is a mismatch between the number of 1s in a character and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the value of the parity bit. When an address bit is included in the character,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>it is included in the parity calculation. When a parity error is detected,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the UCPE bit is set.</td>
</tr>
<tr>
<td>Receive overrun</td>
<td>UCOE</td>
<td>An overrun error occurs when a character is loaded into UCxRXBUF before the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>prior character has been read. When an overrun occurs, the UCOE bit is set.</td>
</tr>
<tr>
<td>Break condition</td>
<td>UCBRK</td>
<td>When not using automatic baud-rate detection, a break is detected when all</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data, parity, and stop bits are low. When a break condition is detected,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the UCBRK bit is set. A break condition can also set the interrupt flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UCRXIFG if the break interrupt enable UCBRKIE bit is set.</td>
</tr>
</tbody>
</table>
UART

• MSP432 UART Implementation
  • Clock setup for desired Baud Rate

Baud-rate settings quick set up

To calculate \( N = f_{BRCLK} / \text{baud rate} \) [if \( N > 16 \) continue with step 3, otherwise with step 2]
1. \( OS16 = 0, UCBRx = \text{INT}(N) \) [continue with step 4]
2. \( OS16 = 1, UCBRx = \text{INT}(N/16), UCBRFx = \text{INT}([N/16] - \text{INT}(N/16)) \times 16) \)
3. \( UCBRSx \) can be found by looking up the fractional part of \( N = N - \text{INT}(N) \) in table Table 22-4
4. If \( OS16 = 0 \) was chosen, a detailed error calculation is recommended to be performed

Table 22-4. UCBRSx Settings for Fractional Portion of \( N = f_{BRCLK}/\text{Baud Rate} \)

<table>
<thead>
<tr>
<th>Fractional Portion of N</th>
<th>UCBRSx((^{(1)}))</th>
<th>Fractional Portion of N</th>
<th>UCBRSx((^{(1)}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>0x00</td>
<td>0.5002</td>
<td>0xAA</td>
</tr>
<tr>
<td>0.0529</td>
<td>0x01</td>
<td>0.5715</td>
<td>0x6B</td>
</tr>
<tr>
<td>0.0715</td>
<td>0x02</td>
<td>0.6003</td>
<td>0xAD</td>
</tr>
<tr>
<td>0.0835</td>
<td>0x04</td>
<td>0.6254</td>
<td>0x85</td>
</tr>
</tbody>
</table>
# UART

- **MSP432 UART Registers**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Acronym</th>
<th>Register Name</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>00h</td>
<td>UCAXxCTLW0</td>
<td>eUSCI_Ax Control Word 0</td>
<td>Section 22.4.1</td>
</tr>
<tr>
<td>00h</td>
<td>UCAXxCTL1</td>
<td>eUSCI_Ax Control 1</td>
<td></td>
</tr>
<tr>
<td>01h</td>
<td>UCAXxCTL0</td>
<td>eUSCI_Ax Control 0</td>
<td></td>
</tr>
<tr>
<td>02h</td>
<td>UCAXxCTLW1</td>
<td>eUSCI_Ax Control Word 1</td>
<td>Section 22.4.2</td>
</tr>
<tr>
<td>06h</td>
<td>UCAXxBRW</td>
<td>eUSCI_Ax Baud Rate Control Word</td>
<td>Section 22.4.3</td>
</tr>
<tr>
<td>06h</td>
<td>UCAXxBR0</td>
<td>eUSCI_Ax Baud Rate Control 0</td>
<td></td>
</tr>
<tr>
<td>07h</td>
<td>UCAXxBR1</td>
<td>eUSCI_Ax Baud Rate Control 1</td>
<td></td>
</tr>
<tr>
<td>08h</td>
<td>UCAXxMCTLW</td>
<td>eUSCI_Ax Modulation Control Word</td>
<td>Section 22.4.4</td>
</tr>
<tr>
<td>0Ah</td>
<td>UCAXxSTATW</td>
<td>eUSCI_Ax Status</td>
<td>Section 22.4.5</td>
</tr>
<tr>
<td>0Ch</td>
<td>UCAXxRXBUF</td>
<td>eUSCI_Ax Receive Buffer</td>
<td>Section 22.4.6</td>
</tr>
<tr>
<td>0Eh</td>
<td>UCAXxTXBUF</td>
<td>eUSCI_Ax Transmit Buffer</td>
<td>Section 22.4.7</td>
</tr>
<tr>
<td>10h</td>
<td>UCAXxABCTL</td>
<td>eUSCI_Ax Auto Baud Rate Control</td>
<td>Section 22.4.8</td>
</tr>
<tr>
<td>12h</td>
<td>UCAXxIRCTRL</td>
<td>eUSCI_Ax IrDA Control</td>
<td>Section 22.4.9</td>
</tr>
<tr>
<td>12h</td>
<td>UCAXxIRTCTL</td>
<td>eUSCI_Ax IrDA Transmit Control</td>
<td></td>
</tr>
<tr>
<td>13h</td>
<td>UCAXxIRRCTRL</td>
<td>eUSCI_Ax IrDA Receive Control</td>
<td></td>
</tr>
<tr>
<td>1Ah</td>
<td>UCAXxE</td>
<td>eUSCI_Ax Interrupt Enable</td>
<td>Section 22.4.10</td>
</tr>
<tr>
<td>1Ch</td>
<td>UCAXxFG</td>
<td>eUSCI_Ax Interrupt Flag</td>
<td>Section 22.4.11</td>
</tr>
<tr>
<td>1Eh</td>
<td>UCAXxIV</td>
<td>eUSCI_Ax Interrupt Vector</td>
<td>Section 22.4.12</td>
</tr>
</tbody>
</table>
# UART

- **MSP432 UART Control Word 0 Register**

### Figure 22-12. UCxCTLW0 Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>UCPEN</td>
<td>RW</td>
<td>0h</td>
<td>Parity enable. 0b = Parity disabled, 1b = Parity enabled. Parity bit is generated (UCxTXD) and expected (UCxRXD). In address-bit multiprocessor mode, the address bit is included in the parity calculation.</td>
</tr>
<tr>
<td>14</td>
<td>UCPAR</td>
<td>RW</td>
<td>0h</td>
<td>Parity select. UCPAR is not used when parity is disabled. 0b = Odd parity, 1b = Even parity</td>
</tr>
<tr>
<td>13</td>
<td>UCMSB</td>
<td>RW</td>
<td>0h</td>
<td>MSB first select. Controls the direction of the receive and transmit shift register. 0b = LSB first, 1b = MSB first</td>
</tr>
<tr>
<td>12</td>
<td>UC7BIT</td>
<td>RW</td>
<td>0h</td>
<td>Character length. Selects 7-bit or 8-bit character length. 0b = 8-bit data, 1b = 7-bit data</td>
</tr>
</tbody>
</table>
### UART

- **MSP432 UART Control Word 0 Register**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
</table>
| 11  | UCSPB    | RW   | 0h    | Stop bit select. Number of stop bits.  
0b = One stop bit  
1b = Two stop bits |
| 10-9| UCMODEx  | RW   | 0h    | eUSCI_A mode. The UCMODEx bits select the asynchronous mode when UCSYNC = 0.  
00b = UART mode  
01b = Idle-line multiprocessor mode  
10b = Address-bit multiprocessor mode  
11b = UART mode with automatic baud-rate detection |
| 8   | UCSYNC   | RW   | 0h    | Synchronous mode enable  
0b = Asynchronous mode  
1b = Synchronous mode |
| 7-6 | UCSSELx  | RW   | 0h    | eUSCI_A clock source select. These bits select the BRCLK source clock.  
00b = UCLK  
01b = ACLK  
10b = SMCLK  
11b = SMCLK |
| 5   | UCRXEIE  | RW   | 0h    | Receive erroneous-character interrupt enable  
0b = Erroneous characters rejected and UCRXIFG is not set.  
1b = Erroneous characters received set UCRXIFG. |
| 4   | UCBRKIE  | RW   | 0h    | Receive break character interrupt enable  
0b = Received break characters do not set UCRXIFG.  
1b = Received break characters set UCRXIFG. |
## UART

- **MSP432 UART Control Word 0 Register**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3   | UCDORM   | RW   | 0h    | Dormant. Puts eUSCI_A into sleep mode.  
0b = Not dormant. All received characters set UCRXIFG.  
1b = Dormant. Only characters that are preceded by an idle-line or with address bit set UCRXIFG. In UART mode with automatic baud-rate detection, only the combination of a break and synch field sets UCRXIFG. |
| 2   | UCTXADDR | RW   | 0h    | Transmit address. Next frame to be transmitted is marked as address, depending on the selected multiprocessor mode.  
0b = Next frame transmitted is data.  
1b = Next frame transmitted is an address. |
| 1   | UCTXBRK  | RW   | 0h    | Transmit break. Transmits a break with the next write to the transmit buffer. In UART mode with automatic baud-rate detection, 055h must be written into UCAXTXBUF to generate the required break/synch fields. Otherwise, 0h must be written into the transmit buffer.  
0b = Next frame transmitted is not a break.  
1b = Next frame transmitted is a break or a break/synch. |
| 0   | UCSWRST  | RW   | 1h    | Software reset enable  
0b = Disabled. eUSCI_A reset released for operation.  
1b = Enabled. eUSCI_A logic held in reset state. |
• MSP432 UART Control Word 1 Register

Figure 22-13. UCAXCTLW1 Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-2</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
</tbody>
</table>
| 1-0  | UCGLITx  | RW   | 3h    | Deglitch time  
00b = Approximately 5 ns  
01b = Approximately 20 ns  
10b = Approximately 30 ns  
11b = Approximately 50 ns |
UART

• MSP432 UART Baud Rate Control Word Register

Figure 22-14. UCAXBRW Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-0</td>
<td>UCBRx</td>
<td>RW</td>
<td>0h</td>
<td>Clock prescaler setting of the baud-rate generator</td>
</tr>
</tbody>
</table>
UART

- MSP432 UART Modulation Control Word Register

### Figure 22-15. UCA\textsubscript{x}MCTLW Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-8</td>
<td>UCBRS\textsubscript{x}</td>
<td>RW</td>
<td>0h</td>
<td>Second modulation stage select. These bits hold a free modulation pattern for BITCLK.</td>
</tr>
<tr>
<td>7-4</td>
<td>UCBRF\textsubscript{x}</td>
<td>RW</td>
<td>0h</td>
<td>First modulation stage select. These bits determine the modulation pattern for BITCLK\textsubscript{16} when UCOS\textsubscript{16} = 1. Ignored with UCOS\textsubscript{16} = 0. The &quot;Oversampling Baud-Rate Generation&quot; section shows the modulation pattern.</td>
</tr>
<tr>
<td>3-1</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
</tbody>
</table>
| 0    | UCOS\textsubscript{16} | RW   | 0h    | Oversampling mode enabled  
0b = Disabled  
1b = Enabled |
UART

• MSP432 UART Status Word Register

Figure 22-16. UCxSTATW Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-8</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
<tr>
<td>7</td>
<td>UCLISTEN</td>
<td>RW</td>
<td>0h</td>
<td>Listen enable. The UCLISTEN bit selects loopback mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0b = Disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1b = Enabled. UCxTXD is internally fed back to the receiver.</td>
</tr>
<tr>
<td>6</td>
<td>UCFE</td>
<td>RW</td>
<td>0h</td>
<td>Framing error flag. UCFE is cleared when UCxRXBUF is read.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0b = No error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1b = Character received with low stop bit</td>
</tr>
<tr>
<td>5</td>
<td>UCOE</td>
<td>RW</td>
<td>0h</td>
<td>Overrun error flag. This bit is set when a character is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>transferred into UCxRXBUF before the previous character was</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>read. UCOE is cleared automatically when UCxRXBUF is read,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>and must not be cleared by software.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Otherwise, it does not function correctly.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0b = No error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1b = Overrun error occurred.</td>
</tr>
</tbody>
</table>
### UART

- **MSP432 UART Status Word Register**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>UCPE</td>
<td>RW</td>
<td>0h</td>
<td>Parity error flag. When UCPEN = 0, UCPE is read as 0. UCPE is cleared when UCAxRXBUF is read. 0b = No error, 1b = Character received with parity error</td>
</tr>
<tr>
<td>3</td>
<td>UCBRK</td>
<td>RW</td>
<td>0h</td>
<td>Break detect flag. UCBRK is cleared when UCAxRXBUF is read. 0b = No break condition, 1b = Break condition occurred.</td>
</tr>
<tr>
<td>2</td>
<td>UCRXERR</td>
<td>RW</td>
<td>0h</td>
<td>Receive error flag. This bit indicates a character was received with one or more errors. When UCRXERR = 1, one or more error flags, UCFE, UCPE, or UCOE is also set. UCRXERR is cleared when UCAxRXBUF is read. 0b = No receive errors detected, 1b = Receive error detected</td>
</tr>
<tr>
<td>1</td>
<td>UCADDR UCIDL</td>
<td>RW</td>
<td>0h</td>
<td>UCADDR: Address received in address-bit multiprocessor mode. UCADDR is cleared when UCAxRXBUF is read. UCIDL: Idle line detected in idle-line multiprocessor mode. UCIDL is cleared when UCAxRXBUF is read. 0b = UCADDR: Received character is data. UCIDL: No idle line detected, 1b = UCADDR: Received character is an address. UCIDL: Idle line detected</td>
</tr>
<tr>
<td>0</td>
<td>UCBUSY</td>
<td>R</td>
<td>0h</td>
<td>eUSCI_A busy. This bit indicates if a transmit or receive operation is in progress. 0b = eUSCI_A inactive, 1b = eUSCI_A transmitting or receiving</td>
</tr>
</tbody>
</table>
UART

- MSP432 UART Receive Buffer Register

**Figure 22-17. UCAXRXBUF Register**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-8</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
<tr>
<td>7-0</td>
<td>UCRXBUFx</td>
<td>R</td>
<td>0h</td>
<td>The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCAXRXBUF resets the receive-error bits, the UCADDR or UCIDLE bit, and UCRXIFG. In 7-bit data mode, UCAXRXBUF is LSB justified and the MSB is always reset.</td>
</tr>
</tbody>
</table>
UART

- MSP432 UART Transmit Buffer Register

![UART Transmit Buffer Register](image)

**Table 22-14. UCAxTXBUF Register Description**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-8</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
<tr>
<td>7-0</td>
<td>UCTXBUFx</td>
<td>RW</td>
<td>0h</td>
<td>The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted on UCAxTXD. Writing to the transmit data buffer clears UCTXIFG. The MSB of UCAxTXBUF is not used for 7-bit data and is reset.</td>
</tr>
</tbody>
</table>
### UART

- **MSP432 UART Auto Baud Rate Control Register**

#### Table 22-15. UCxABCTL Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-6</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
</tbody>
</table>
| 5-4 | UCDELIMx | RW   | 0h    | Break/synch delimiter length  
0b = 1 bit time  
01b = 2 bit times  
10b = 3 bit times  
11b = 4 bit times |
| 3   | UCSTOE   | RW   | 0h    | Synch field time out error  
0b = No error  
1b = Length of synch field exceeded measurable time. |
| 2   | UCBTOE   | RW   | 0h    | Break time out error  
0b = No error  
1b = Length of break field exceeded 22 bit times. |
| 1   | Reserved | R    | 0h    | Reserved                                                                   |
| 0   | UCABDEN  | RW   | 0h    | Automatic baud-rate detect enable  
0b = Baud-rate detection disabled. Length of break and synch field is not measured.  
1b = Baud-rate detection enabled. Length of break and synch field is measured and baud-rate settings are changed accordingly. |
UART

- MSP432 UART IRDA Control Register

![Table 22-16. UCAxIRCTL Register Description](image)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-10</td>
<td>UCIRRXFLx</td>
<td>RW</td>
<td>0h</td>
<td>Receive filter length. The minimum pulse length for receive is given by: ( t_{\text{MIN}} = (\text{UCIRRXFLx} + 4) / (2 \times f_{\text{RTXCLK}}) )</td>
</tr>
<tr>
<td>9</td>
<td>UCIRRXPL</td>
<td>RW</td>
<td>0h</td>
<td>IrDA receive input UCAXRXD polarity 0b = IrDA transceiver delivers a high pulse when a light pulse is seen. 1b = IrDA transceiver delivers a low pulse when a light pulse is seen.</td>
</tr>
<tr>
<td>8</td>
<td>UCIRRXFE</td>
<td>RW</td>
<td>0h</td>
<td>IrDA receive filter enabled 0b = Receive filter disabled 1b = Receive filter enabled</td>
</tr>
<tr>
<td>7-2</td>
<td>UCIRTXPLx</td>
<td>RW</td>
<td>0h</td>
<td>Transmit pulse length. Pulse length ( t_{\text{PULSE}} = (\text{UCIRTXPLx} + 1) / (2 \times f_{\text{RTXCLK}}) )</td>
</tr>
<tr>
<td>1</td>
<td>UCIRTXCLK</td>
<td>RW</td>
<td>0h</td>
<td>IrDA transmit pulse clock select 0b = BRCLK 1b = BITCLK16 when UCOS16 = 1. Otherwise, BRCLK.</td>
</tr>
<tr>
<td>0</td>
<td>UCIREN</td>
<td>RW</td>
<td>0h</td>
<td>IrDA encoder and decoder enable 0b = IrDA encoder/decoder disabled 1b = IrDA encoder/decoder enabled</td>
</tr>
</tbody>
</table>
UART

- MSP432 UART Interrupt Enable Register

![UART Interrupt Enable Register Diagram](image)

### Table 22-17. UCAxIE Register Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-4</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
<tr>
<td>3</td>
<td>UCTXCPTIE</td>
<td>RW</td>
<td>0h</td>
<td>Transmit complete interrupt enable&lt;br&gt;0b = Interrupt disabled&lt;br&gt;1b = Interrupt enabled</td>
</tr>
<tr>
<td>2</td>
<td>UCSTTIE</td>
<td>RW</td>
<td>0h</td>
<td>Start bit interrupt enable&lt;br&gt;0b = Interrupt disabled&lt;br&gt;1b = Interrupt enabled</td>
</tr>
<tr>
<td>1</td>
<td>UCTXIE</td>
<td>RW</td>
<td>0h</td>
<td>Transmit interrupt enable&lt;br&gt;0b = Interrupt disabled&lt;br&gt;1b = Interrupt enabled</td>
</tr>
<tr>
<td>0</td>
<td>UCRXIE</td>
<td>RW</td>
<td>0h</td>
<td>Receive interrupt enable&lt;br&gt;0b = Interrupt disabled&lt;br&gt;1b = Interrupt enabled</td>
</tr>
</tbody>
</table>
**UART**

- MSP432 UART Interrupt Flag Register

![Figure 22-22. UCAxIFG Register](image)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-4</td>
<td>Reserved</td>
<td>R</td>
<td>0h</td>
<td>Reserved</td>
</tr>
<tr>
<td>3</td>
<td>UCTXCPTIFG</td>
<td>RW</td>
<td>0h</td>
<td>Transmit complete interrupt flag. UCTXCPTIFG is set when the entire byte in the internal shift register got shifted out and UCAxTXBUF is empty. 0b = No interrupt pending 1b = Interrupt pending</td>
</tr>
<tr>
<td>2</td>
<td>UCSTTIFG</td>
<td>RW</td>
<td>0h</td>
<td>Start bit interrupt flag. UCSTTIFG is set after a Start bit was received 0b = No interrupt pending 1b = Interrupt pending</td>
</tr>
<tr>
<td>1</td>
<td>UCTXIFG</td>
<td>RW</td>
<td>1h</td>
<td>Transmit interrupt flag. UCTXIFG is set when UCAxTXBUF empty. 0b = No interrupt pending 1b = Interrupt pending</td>
</tr>
<tr>
<td>0</td>
<td>UCRXIFG</td>
<td>RW</td>
<td>0h</td>
<td>Receive interrupt flag. UCRXIFG is set when UCAxRXBUF has received a complete character. 0b = No interrupt pending 1b = Interrupt pending</td>
</tr>
</tbody>
</table>
**UART**

- MSP432 UART Interrupt Vector Register

![UART Interrupt Vector Register](image)

**Figure 22-23. UCAXIV Register**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Type</th>
<th>Reset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-0</td>
<td>UCIVx</td>
<td>R</td>
<td>0h</td>
<td>eUSCI_A interrupt vector value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00h = No interrupt pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>02h = Interrupt Source: Receive buffer full; Interrupt Flag: UCRXIFG; Interrupt Priority: Highest</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>04h = Interrupt Source: Transmit buffer empty; Interrupt Flag: UCTXIFG</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>06h = Interrupt Source: Start bit received; Interrupt Flag: UCSTTIFG</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>08h = Interrupt Source: Transmit complete; Interrupt Flag: UCTXCPTIFG; Interrupt Priority: Lowest</td>
</tr>
</tbody>
</table>
# UART

- Loop-back Example

```c
/*
utart.c
*
*  Created on: Oct 12, 2017
*      Author: johnsontimoj
*/

// uart test file
// Loop back from A1 to A2

#include <stdio.h>
#include <stdint.h>
#include "msp432.h"
#include "msoe_lib_all.h"

void pin_setup(void);
void uart_tx_setup(void);
void uart_rx_setup(void);
void uart_A1_transmit(uint8_t data);

// Global Variable for Interrupts
uint8_t rx_data;
uint8_t tx_data;
uint8_t num_bytes;

int main(void){
    // setup counter
    tx_data = 0;

    // Configure pins
    pin_setup();

    // Setup TWIs
    uart_tx_setup();
    uart_rx_setup();

    while(1){
        // start transmit
        uart_A1_transmit(tx_data);

        // wait for transfer to complete
        Delay_3MHz_ms(50);

        // Print rcvd values
        printf("Tx UART sent %i - Rx UART received %i\n", tx_data, rx_data);

        // Update count and wait
        tx_data++;
    }
    return 0;
} // end main
```
void pin_setup(void)
{
    // Setup USCIA1 transmit
    // Setup USCIA2 Receive
    // Transfer in a while loop
    //
    // USCIA1 --> P2.2-RX, P2.3-TX
    // USCIA2 --> P3.2-RX, P3.3-TX
    //
    P2->SEL0 |= 0x0C;  // Set to USC mode
    P2->SEL1 &= ~0x0C; // 0-1 mode
    P3->SEL0 |= 0x0C;
    P3->SEL1 &= ~0x0C;

    // Nothing else needed - the module
    // sets directions
    return;
} // end pin_setup
UART

```c
void uart_rx_setup(void){
    // USCIA2 - receiver
    // 8 bit, Uart mode, even parity
    //
    // CTLW0 - allow changes
    // sw reset
    EUSCI_A2->CTLW0 = 0x0001;
    //
    // CLTW0
    // P E LSB 8b 1st uart Async Aclk noche nobrkie awake noaddr nobrk rst
    // 1 1 0 0 0 00 0 01 0 0 0 0 0 0 0 1
    // C041
    EUSCI_A2->CTLW0 = 0xC041;
    //
    // No Deglitch
    //
    // BRW
    // /64 - 0000 0000 0100 0000
    // 0040
    EUSCI_A2->BRW = 0x0040;
    //
    // No Modulation
    //
    // No Auto baud rate
    //
    // No IrDA
    //
    // CTLW0
    // release reset
    EUSCI_A2->CTLW0 &= ~0x01;
    //
    // IE
    //
    // TC st TXbuf RC
    // x x x x x x x 0 0 0 0 1
    // 0002
    EUSCI_A2->IE = 0x0001;
    //
    // NVIC
    // EUSCI_A2 is int 18
    //
    NVIC->IP[18] = 0x60; // priority = 3
    NVIC->ISER[0] |= 0x00040000; // enable

    return;
} // end uart_rx_setup

void uart_A1_transmit(uint8_t data){
    // Check for busy
    while(EUSCI_A1->STATW & 0x0001) ;
    // tx buffer
    EUSCI_A1->TXBUF = data;

    return;
} // end uart_A1_transmit

void EUSCIA1_IRQHandler(void){
    // Check for TX buffer empty (tx complete)
    if(EUSCI_A1->IFG & 0x0002){
        // clear flag
        EUSCI_A1->IFG &= ~0x0002;
    }
} // end uart tx int handler

void EUSCIA2_IRQHandler(void){
    // Check for RX flag (receive complete)
    if(EUSCI_A2->IFG & 0x0001){
        EUSCI_A2->IFG &= ~0x0001;
        // clear flag
        rx_data = EUSCI_A2->RXBUF; // copy data
    }
} // end uart rx int handler
```
UART

- MSP432 UART
- Examples