Why use a UART?

- UARTs are everywhere!
- Simple way to send data from one system to another system
- Add additional functionality to an application
Why use a UART from Exar?

- Largest and broadest UART portfolio
- Highest performance UARTs
- UARTs with the most enhanced features
- Excellent technical support
- Exar also has serial transceivers!
What is a UART?

- **Universal Asynchronous Receiver/Transmitter**
- **Traditional Definition**: Converts parallel (8-bit) data to serial data and vice versa
What is a UART?
## 16550 UART Registers

<table>
<thead>
<tr>
<th>Address A2-A0</th>
<th>Register Name</th>
<th>Read/Write</th>
<th>Register Function</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>DLL – Divisor LSB</td>
<td>Write-Only</td>
<td>Divisor (LSB) for BRG</td>
<td>LCR bit-7 = 1</td>
</tr>
<tr>
<td>0 0 1</td>
<td>DLM – Divisor MSB</td>
<td>Read-Only</td>
<td>Divisor (MSB) for BRG</td>
<td>LCR bit-7 = 1</td>
</tr>
<tr>
<td>0 0 0</td>
<td>THR – Transmit Holding Register</td>
<td>Write-Only</td>
<td>Loading data into TX FIFO</td>
<td>LCR bit-7 = 0</td>
</tr>
<tr>
<td>0 0 0</td>
<td>RHR – Receive Holding Register</td>
<td>Read-Only</td>
<td>Unloading data from RX FIFO</td>
<td>LCR bit-7 = 0</td>
</tr>
<tr>
<td>0 0 1</td>
<td>IER – Interrupt Enable Register</td>
<td>Read/Write</td>
<td>Enable interrupts</td>
<td></td>
</tr>
<tr>
<td>0 1 0</td>
<td>FCR – FIFO Control Register</td>
<td>Write-Only</td>
<td>FIFO enable and reset</td>
<td></td>
</tr>
<tr>
<td>0 1 0</td>
<td>LSR – Interrupt Status Register</td>
<td>Read-Only</td>
<td>Status of highest priority interrupt</td>
<td></td>
</tr>
<tr>
<td>0 1 1</td>
<td>LCR – Line Control Register</td>
<td>Read/Write</td>
<td>Word length, stop bits, parity select, send break,</td>
<td></td>
</tr>
<tr>
<td>1 0 0</td>
<td>MCR – Modem Control Register</td>
<td>Read/Write</td>
<td>select divisor registers</td>
<td>RTS# and DTR# output control</td>
</tr>
<tr>
<td>1 0 1</td>
<td>LSR – Line Status Register</td>
<td>Read-Only</td>
<td>RX Errors/Status TX Status</td>
<td>Interrupt output enable</td>
</tr>
<tr>
<td>1 1 0</td>
<td>MSR – Modem Status Register</td>
<td>Read-Only</td>
<td>Modem Input Status</td>
<td>Internal Loopback enable</td>
</tr>
<tr>
<td>1 1 1</td>
<td>SPR – Scratch Pad Register</td>
<td>Read/Write</td>
<td>General Purpose Register</td>
<td></td>
</tr>
</tbody>
</table>
# 16550 UART Registers

<table>
<thead>
<tr>
<th>Address A2-A0</th>
<th>Register Name</th>
<th>R/W</th>
<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 0 0</td>
<td>DLL</td>
<td>R/W</td>
<td>Bit-7</td>
<td>Bit-6</td>
<td>Bit-5</td>
<td>Bit-4</td>
<td>Bit-3</td>
<td>Bit-2</td>
<td>Bit-1</td>
<td>Bit-0</td>
</tr>
<tr>
<td>0 0 1</td>
<td>DLM</td>
<td>R/W</td>
<td>Bit-7</td>
<td>Bit-6</td>
<td>Bit-5</td>
<td>Bit-4</td>
<td>Bit-3</td>
<td>Bit-2</td>
<td>Bit-1</td>
<td>Bit-0</td>
</tr>
<tr>
<td>0 0 0</td>
<td>THR</td>
<td>W</td>
<td>Bit-7</td>
<td>Bit-6</td>
<td>Bit-5</td>
<td>Bit-4</td>
<td>Bit-3</td>
<td>Bit-2</td>
<td>Bit-1</td>
<td>Bit-0</td>
</tr>
<tr>
<td>0 0 0</td>
<td>RHR</td>
<td>R</td>
<td>Bit-7</td>
<td>Bit-6</td>
<td>Bit-5</td>
<td>Bit-4</td>
<td>Bit-3</td>
<td>Bit-2</td>
<td>Bit-1</td>
<td>Bit-0</td>
</tr>
<tr>
<td>0 0 1</td>
<td>IER</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MSR</td>
<td>LSR</td>
</tr>
<tr>
<td>0 1 0</td>
<td>FCR</td>
<td>W</td>
<td>RX Trig Level</td>
<td>RX Trig Level</td>
<td>0</td>
<td>0</td>
<td>DMA Mode</td>
<td>TX FIFO Reset</td>
<td>RX FIFO Reset</td>
<td>FIFO Enable</td>
</tr>
<tr>
<td>0 1 0</td>
<td>ISR</td>
<td>R</td>
<td>FIFOs Enabled</td>
<td>FIFOs Enabled</td>
<td>0</td>
<td>0</td>
<td>INT Source</td>
<td>INT Source</td>
<td>INT Source</td>
<td>INT Source</td>
</tr>
<tr>
<td>0 1 1</td>
<td>LCR</td>
<td>R/W</td>
<td>Divisor Enable</td>
<td>Set TX Break</td>
<td>Set Parity</td>
<td>Even Parity</td>
<td>Parity Enable</td>
<td>Stop Bits</td>
<td>Word Length</td>
<td>Word Length</td>
</tr>
<tr>
<td>1 0 0</td>
<td>MCR</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Internal Loopback</td>
<td>INT / OP2#</td>
</tr>
<tr>
<td>1 0 1</td>
<td>LSR</td>
<td>R</td>
<td>RX FIFO Error</td>
<td>THR/TSR Empty</td>
<td>THR Empty</td>
<td>RX Break</td>
<td>RX Framing</td>
<td>RX Parity</td>
<td>RX Overrun</td>
<td>RX Data Ready</td>
</tr>
<tr>
<td>1 1 0</td>
<td>MSR</td>
<td>R</td>
<td>CD#</td>
<td>RI#</td>
<td>DSR#</td>
<td>CTS#</td>
<td>Delta CD#</td>
<td>Delta RI#</td>
<td>Delta DSR#</td>
<td>Delta CTS#</td>
</tr>
<tr>
<td>1 1 1</td>
<td>SPR</td>
<td>R/W</td>
<td>Bit-7</td>
<td>Bit-6</td>
<td>Bit-5</td>
<td>Bit-4</td>
<td>Bit-3</td>
<td>Bit-2</td>
<td>Bit-1</td>
<td>Bit-0</td>
</tr>
</tbody>
</table>
Baud Rate Generator (BRG)

- Used to generate the baud rates for both the transmitter and receiver
- Not required for any other function including reads and writes
- Crystal or External Clock
- 16-bit divisor programmed in DLM/DLL registers
Baud Rate Generator (BRG)

Baud Rate = \frac{\text{Clock Frequency}}{(\text{Sampling Rate}) \times (\text{Divisor})}

- **Standard clock frequencies are multiples of 1.8432 MHz**
  - 3.6864 MHz, 7.3728 MHz, 14.7456 MHz, 18.432 MHz, 22.1184 MHz
- **Standard baud rates are multiples of 9600 bps**
  - 19200 bps, 38400 bps, 57600 bps, 115200 bps, 230400 bps, 460800 bps, 921600 bps
- **Sampling rate is 16**
- **Divisor values are written into the DLM and DLL registers**
  - Divisor values are 1 to \((2^{16} - 1)\) in increments of 1

Baud Rate = \frac{14.7456 \text{ MHz}}{(16) \times (1)} = 921600 \text{ bps}
Transmitter

- Parallel-to-serial conversion
- Non-FIFO Mode
  - Transmit Holding Register (THR) and Transmit Shift Register (TSR)
- FIFO Mode
  - Transmit (TX) FIFO and Transmit Shift Register (TSR)
- 16X timing for bit shifting
- Character Framing
- Parity Insertion
- TX FIFO interrupt and status
Transmitter – Non-FIFO mode

- Write Data to Transmit Holding Register (THR)
- Data in THR is transferred to Transmit Shift Register (TSR) when TSR is empty
- TSR shifts the data out on the TX output pin
Transmitter – FIFO Mode

- Write Data to Transmit Holding Register (THR)
- Transmit data is queued in TX FIFO
- Data in TX FIFO is transferred to Transmit Shift Register (TSR) when TSR is empty
- TSR shifts data out on TX output pin
TX Character Framing

- **Start Bit**
- **Data Bits of 5, 6, 7 or 8**
- **Parity Bit**
- **Stop Bit of 1, 1.5 or 2**
- **Example:**
  - Start, 8 data, parity, with 1 stop bit

TX | Idle | T | P | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | S |
---|------|---|---|----|----|----|----|----|----|----|----|---|
Idle = “Mark” or “1”
Receiver

• Serial-to-Parallel Conversion
• Non-FIFO Mode
  • Receive Holding Register (RHR) and Receive Shift Register (RSR)
• FIFO Mode
  • RX FIFO and RSR
• 16X timing clock for mid bit sampling
• Start bit detection and verification
• RX FIFO is 11 bits wide
  • 8 data bits
  • 3 error bits or error tags
Receiver – Non-FIFO Mode

- Incoming data is received in the Receive Shift Register (RSR)
- Received data is transferred to the RHR
- Error tags associated with data in RHR can be read via LSR
- Read RHR to read the data out
Receiver – FIFO Mode

- Incoming data is received in the Receive Shift Register (RSR)
- Received data is queued in the RX FIFO
- Error tags associated with data in RHR can be read via LSR
- Read RHR to read the data out
RX Character Validation

- **Start bit detection and validation**
  - HIGH to LOW transition indicates a start bit
  - Start bit validated if RX input is still LOW during mid bit sampling
- **Data, parity and stop bits are sampled at mid bit**
- **A valid stop bit is HIGH when the stop bit is sampled**

<table>
<thead>
<tr>
<th>Idle</th>
<th>T</th>
<th>P</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>S</th>
</tr>
</thead>
</table>

Idle = “Mark” or “1”
RX Error Reporting

• **Line Status errors**
  • Error tags are associated with each byte
    • Framing error if stop bit is not detected
    • Parity error if parity bit is incorrect
    • Break detected if RX input is LOW for duration of one character time and stop bit is not detected
  • Overrun error if character is received in RSR when RX FIFO is full
    • Non-FIFO mode
      • RHR has a data byte and data received in RSR
      • RSR data overwrites RHR data
    • FIFO mode
      • RX FIFO is full and data is received in RSR
      • Data in RX FIFO is not overwritten by data in RSR
Modem I/Os

- **Legacy Modem Signals**

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Description</th>
<th>Input/Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTS#</td>
<td>Request-to-Send</td>
<td>Output</td>
</tr>
<tr>
<td>CTS#</td>
<td>Clear-to-Send</td>
<td>Input</td>
</tr>
<tr>
<td>DTR#</td>
<td>Data-Terminal-Ready</td>
<td>Output</td>
</tr>
<tr>
<td>DSR#</td>
<td>Data-Set-Ready</td>
<td>Input</td>
</tr>
<tr>
<td>CD#</td>
<td>Carrier-Detect</td>
<td>Input</td>
</tr>
<tr>
<td>RI#</td>
<td>Ring-Indicator</td>
<td>Input</td>
</tr>
</tbody>
</table>

- Used for hardware flow control or as general purpose inputs or outputs
Internal Loopback Mode

Transmit Shift Register (THR/FIFO)

Receive Shift Register (RHR/FIFO)

Modem / General Purpose Control Logic

Internal Data Bus Lines and Control Signals

TX

RX

RTS#

MCR bit-4=1

CTS#

DTR#

DSR#

RI#

CD#

VCC

VCC
Interrupts

<table>
<thead>
<tr>
<th>Priority Level</th>
<th>ISR bit-3</th>
<th>ISR bit-2</th>
<th>ISR bit-1</th>
<th>ISR bit-0</th>
<th>Source of Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>LSR (RX Data Error)</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>RXRDY (RX Data Time-out)</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>RXRDY (RX Data Ready)</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>TXRDY (TX Empty)</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MSR (Modem Status)</td>
</tr>
<tr>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>None</td>
</tr>
</tbody>
</table>

• **Interrupt Source Register (ISR)**
  • If there are multiple interrupts, ISR reports only the highest pending interrupt
  • Lower priority interrupts will be reported when higher priority interrupts are cleared
E-mail hotline: uarttechsupport@exar.com