# BB16CF950+

## CompactFlash/PCMCIA to BlueCore Interface IC

**Databook version 1.04**

# Table of Contents



# 1. Description

## 1.1. General

The BB16CF950+ is a UART (Universal Asynchronous Receiver-Transmitter) with a host interface suitable for direct connection to a CompactFlash or 16-bit PC Card bus. Once installed and configured by the host OS, it provides an eight-byte programming interface which may be configured to be identical to that of the TL16C750 UART from Texas Instruments. It can be configured to fit the requirements of RS232, RS422/485, or Bluetooth™ applications.

## 1.2. Key features

- Glueless interface to CompactFlash or 16-bit PC Card (PCMCIA) bus
- UART is register and functionally compatible with TL16C750
- Compatible with existing 16C750/550/450 device drivers
- Wide supply voltage range 3.0-5.5V
- Compliant with stringent CompactFlash I/O requirements
- Low-power design
- Configuration and CIS data is held in a small, cheap serial EEPROM
- UART enhancements:
    - 128-deep receive and transmit FIFOs
    - Readable FIFO levels and tuneable trigger levels improve device driver performance
    - Programmable "synchronisation factor" allows baud rates up to $f_{clock}/4$
    - Automatic transmitter/receiver enable control capability for RS485 half-duplex applications
    - Extensions to standard register set are implemented in a safe, easy-to-use way
- Added features for Bluetooth designs:
    - Designed to support Bluetooth ICs from CSR's BlueCore range
    - Low-voltage interface pins eliminate the need for level-shifting circuitry
    - SPI mode for factory-programming of BlueCore flash memory
    - Power control can shut down power to Bluetooth circuitry when not in use
    - Automatic reset generation after Bluetooth power-up

## 1.3. The UART

The BB16CF950+ is register and functionally compatible with TL16C750. The TL16C750 is a backwards-compatible upgrade of the TL16C550, and the TL16C550 is a backwards-compatible upgrade of the TL16C450. The TL16C550 is a widely-supported industry-standard UART with 16-deep transmit and receive FIFOs, and many existing device drivers can also use the additional features (such as 64-deep FIFOs, and automatic flow control) of the TL16C750. Device drivers written specially for the BB16CF950+ can make use of added features such as 128-deep FIFOs, readable FIFO levels, and individually programmable FIFO trigger levels.

The BB16CF950+ converts between RS232-format serial data on separate transmit and receive lines, and byte-wide I/O writes and reads on the host interface. Malformed incoming serial data is flagged along with the data in the receive FIFO. The state of the UART can be found at any time by reading status registers, and modem control (handshaking output) lines can be individually controlled.

Although polled-mode operation is possible, the UART will usually be operated on a host-interrupt basis. The interrupt system is designed to allow efficient handling of interrupt service requests from the UART, for example by using the prioritised interrupt identification register, readable FIFO levels, and tunable FIFO trigger levels.

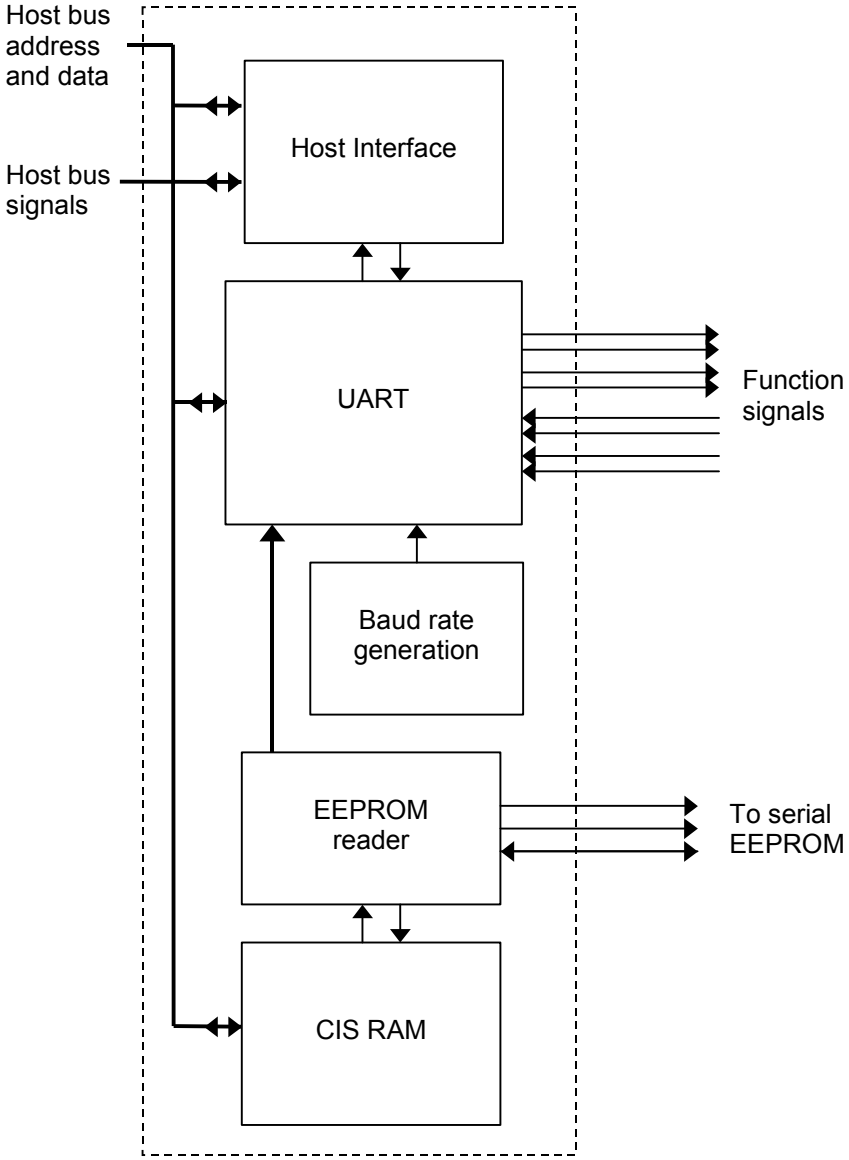
The internal transmitter and receiver logic runs at a programmable synchronisation factor of 4x, 8x, or 16x the serial baud rate. This internal lock is generated by dividing a reference clock by and integer divisor from 1 to $(2^{16} - 1)$. In this way the UART can accommodate a serial rate of up to 5 500 000 baud (using a 22 MHz input clock).

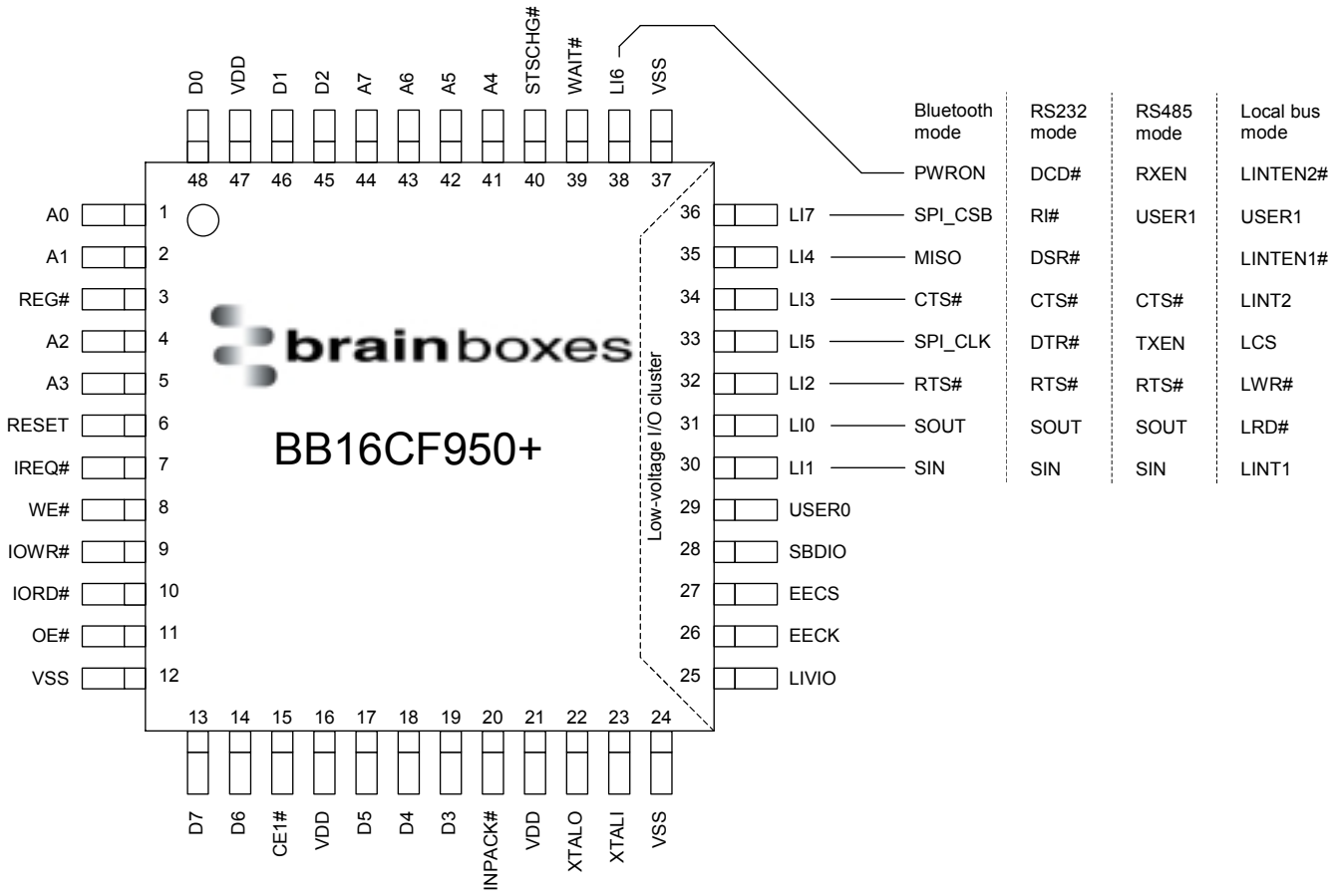
### 1.4. The host interface

The BB16CF950+ provides a host interface that can be directly connected to a CompactFlash or 16-bit PC Card (PCMCIA) bus. It responds to attribute memory accesses for CIS reads and configuration, and once configured it also responds to I/O accesses for control of the UART. The data for the CIS (Card Information Structure) is read from a small external serial EEPROM at start-up, together with information on how the BB16CF950+ should be set up.

# 2. Functional block diagram



# 3. Terminal functions




| Name | Pin | Bus signal class | Description |
|---|---|---|---|
| PC Card/CF+ bus | | | |
| A0 | 1 | Address | Address bus. |
| A1 | 2 | Address | During attribute memory accesses, A7:0 are used to select the CIS data byte or a Function Control Register. |
| A2 | 4 | Address | |
| A3 | 5 | Address | During I/O accesses, A0–A2 are used to select the UART register to be accessed. |
| A4 | 41 | Address | |
| A5 | 42 | Address | |
| A6 | 43 | Address | |
| A7 | 44 | Address | |
| D0 | 48 | Data Bus | Data bus. Eight data lines with 3-state outputs provide a bidirectional path for data between the UART and the host. |
| D1 | 46 | Data Bus | |
| D2 | 45 | Data Bus | |
| D3 | 19 | Data Bus | |
| D4 | 18 | Data Bus | |
| D5 | 17 | Data Bus | |
| D6 | 14 | Data Bus | |
| D7 | 13 | Data Bus | |
| RESET | 6 | Control | |
| CE1# | 15 | Control | |

| Name | Pin | Bus signal class | Description |
|---|---|---|---|
| REG# | 3 | Control | Attribute memory select |
| OE# | 11 | Control | Read strobe for memory accesses |
| WE# | 8 | Control | Write strobe for memory accesses |
| IORD# | 10 | Control | Read strobe for I/O accesses |
| IOWR# | 9 | Control | Write strobe for I/O accesses |
| IREQ# READY | 7 | Status | Interrupt request (when card is configured for I/O) and “card ready” signal. |
| WAIT# | 39 | Status | Read/write cycle delay request |
| STSCHG# | 40 | | Signals status change to host system |
| INPACK# | 20 | Status | Input Port Acknowledge |
| Misc IO | | | |
| USER0 | 29 | | General purpose I/O or reset output |
| EEPROM | | | |
| EECS | 27 | | Chip select to EEPROM Microwire interface |
| EECK | 26 | | Clock signal to EEPROM |
| SBDIO | 28 | | Serial data to and from EEPROM |
| Function signals (Bluetooth mode) | | | |
| SOUT | 31 | | Asynchronous serial data output |
| SIN | 30 | | Asynchronous serial data input |
| RTS# | 32 | | “Request To Send” handshaking output |
| CTS# | 34 | | “Clear To Send” handshaking input |
| MISO | 35 | | Serial data from Bluecore SPI interface |
| SPI_CLK | 33 | | Clock signal to Bluecore SPI interface |
| PWRON | 38 | | Voltage regulator on/off control |
| SPI_CSB# | 36 | | Chip select to Bluecore SPI interface |
| Function signals (RS232 mode) | | | |
| SOUT | 31 | | Asynchronous serial data output |
| SIN | 30 | | Asynchronous serial data input |
| RTS# | 32 | | “Request To Send” handshaking output |
| CTS# | 34 | | “Clear To Send” handshaking input |
| DSR# | 35 | | “Data Set Ready” handshaking input |
| DTR# | 33 | | “Data Terminal Ready” handshaking output |
| DCD# | 38 | | “Data Carrier Detect” modem status input |
| RI# | 36 | | “Ring Indicator” modem status input |
| Function signals (RS485 mode) | | | |
| SOUT | 31 | | Asynchronous serial data output |
| SIN | 30 | | Asynchronous serial data input |
| RTS# | 32 | | “Request To Send” handshaking output |
| CTS# | 34 | | “Clear To Send” handshaking input |
| | 35 | | Unused input – provide pull-up |
| TXEN | 33 | | Transmitter enable |
| RXEN | 38 | | Receiver enable |
| USER1 | 36 | | General purpose I/O or clock output |
| Function signals (local bus mode) | | | |
| LRD# | 31 | | Local bus read strobe |
| LINT1 | 30 | | Interrupt input 1 |
| LWR# | 32 | | Local bus write strobe |
| LINT2 | 34 | | Interrupt input 2 |
| LINTEN1# | 35 | | Interrupt enable 1 |
| LCS | 33 | | Local bus chip select |
| LINTEN2# | 38 | | Interrupt enable 2 |
| USER1 | 36 | | General purpose I/O or clock output |

| Name | Pin | Bus signal class | Description |
|---|---|---|---|
| Oscillator | | | |
| XTALI | 23 | | Clock input or input of oscillator driver |
| XTALO | 22 | | Buffered clock output or output of oscillator driver |
| Power | | | |
| VDD | 16, 21, 47 | | 3.1 to 5.5V power input |
| LIVIO | 25 | | I/O supply for pins 26-36 |
| VSS | 12, 24, 37 | | Ground |

# 4. Timing parameters

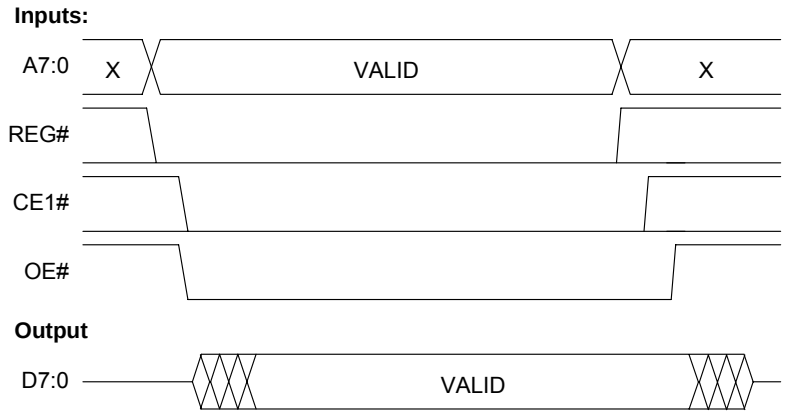
The timings of the host interface are compliant with all requirements of PC Card and CompactFlash specifications.

Timing parameters are classified as follows:

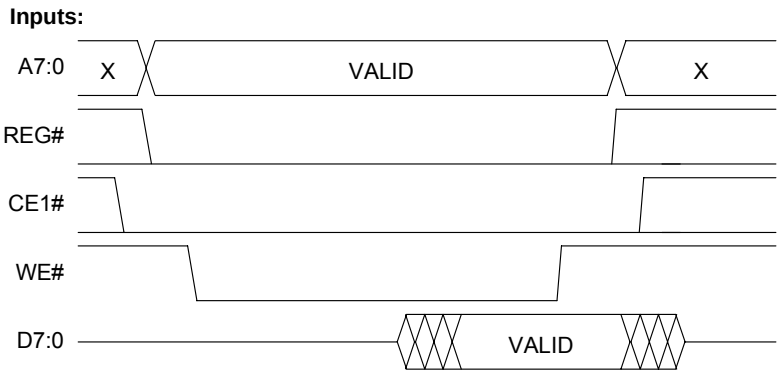
Type G: The BB16CF950+ must generate delays no greater than the “req. max” timings in order to fulfil its purpose. The “act. max” are the maximum delays it will actually generate (from static timing analysis), and these should be smaller than the “req. max” figures.

There is also one instance where the BB16CF950+ must generate delays no smaller than the “req. min” timings in order to fulfil its purpose. The “act. min” are the minimum delays it will actually generate (from static timing analysis), and these should be larger than the “req. min” figures.

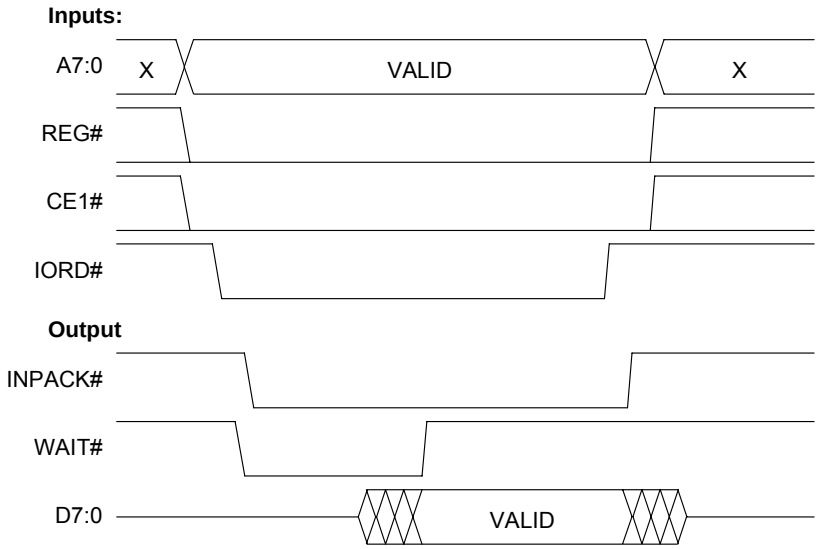
Type A: The BB16CF950+ must accept the “req. min” timings in order to fulfil its purpose. The “act. min” are the minimum timings it can actually accept (from static timing analysis), and these should be smaller than the “req. min” figures.



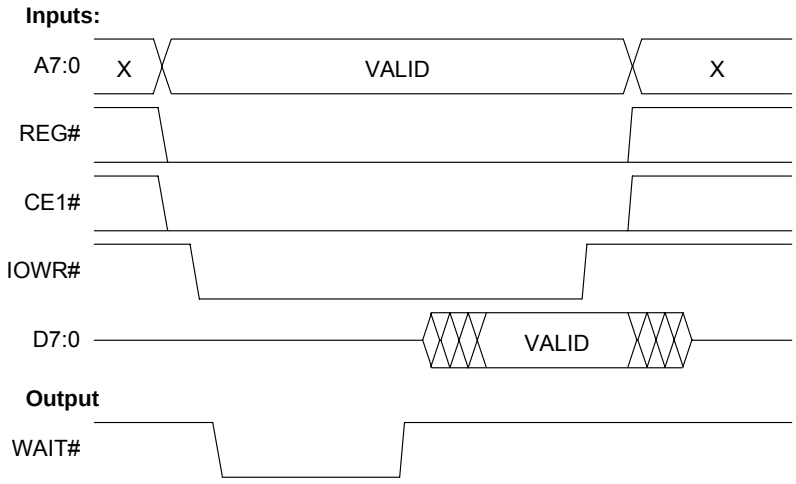

**Attribute memory read cycle**




**Attribute memory write cycle**




**I/O read cycle**




**I/O write cycle**

**Timings of master clock input**

| Type | Signal | Timing | Min | Nom | Max | Unit |
|---|---|---|---|---|---|---|
| A1 | XTALI | Cycle period | 45 | | 71 | ns |
| A1 | XTALI | High time | 3 | | - | ns |

**Timings of host interface bus**

| Type | Signal | Timing | Req. min | Act. min | Req. max | Act. max | Unit |
|---|---|---|---|---|---|---|---|
| A1 | WE# | Cycle period | 250 | | - | - | ns |
| A1 | WE# | Low time | 150 | 3 | - | - | ns |
| A1 | WE# | High time | 60 | | - | - | ns |
| A1 | DATA7:0 | Set-up before WE#↑ | 80 | 3 | - | - | ns |
| A1 | DATA7:0 | Hold after WE#↑ | 30 | 2 | - | - | ns |
| A1 | A7:0, REG# | Set-up before WE#↑ | 180 | 10 | - | - | ns |
| A1 | A7:0, REG# | Hold after WE#↑ | 30 | 1 | - | - | ns |
| A1 | CE1# | Set-up before WE#↑ | 180 | 10 | - | - | |
| A1 | CE1# | Hold after WE#↑ | 20 | -2 | - | - | |
| A1 | IOWR# | Cycle period | 255 | | - | - | ns |
| A1 | IOWR# | Low time | 165 | | - | - | ns |
| A1 | IOWR# | High time | 90 | 3 | - | - | ns |
| A1 | IORD# | Cycle period | 255 | | - | - | ns |
| A1 | IORD# | Low time | 165 | 3 | - | - | ns |
| A1 | IORD# | High time | 90 | | - | - | ns |
| A1 | DATA7:0 | Set-up before IOWR#↑ | 60 | 1 | - | - | ns |
| A1 | DATA7:0 | Hold after IOWR#↑ | 30 | 4 | - | - | ns |
| A1 | A7:0 | Set-up before IOWR#↑ | 235 | 11 | - | - | ns |
| A1 | A7:0 | Hold after IOWR#↑ | 20 | 3 | - | - | ns |
| A1 | REG# | Set-up before IOWR#↑ | 170 | 6 | - | - | ns |
| A1 | REG# | Hold after IOWR#↑ | 0 | 0 | - | - | ns |
| A1 | CE1# | Set-up before IOWR#↑ | 170 | 6 | - | - | ns |
| A1 | CE1# | Hold after IOWR#↑ | 20 | -1 | - | - | ns |
| A1 | A7:0 | Set-up before IORD#↓ | 70 | 17 | - | - | ns |
| A1 | A7:0 | Hold after IORD#↓ | 185 | -2 | - | - | ns |
| A1 | REG# | Set-up before IORD#↓ | 5 | -8 | - | - | ns |
| A1 | REG# | Hold after IORD#↓ | 165 | 13 | - | - | ns |
| A1 | CE1# | Set-up before IORD#↓ | 5 | -8 | - | - | ns |
| A1 | CE1# | Hold after IORD#↓ | 185 | 12 | - | - | ns |
| A1 | RESET | Pulse width (high) | 10 | | - | - | µs |
| G1 | DATA | Propagation delay from A7:0 | - | - | 300 | 212 | ns |
| G1 | DATA | Propagation delay from REG# | - | - | 300 | 33 | ns |
| G1 | DATA | Propagation delay from CE1# | - | - | 300 | 26 | ns |
| G1 | DATA | Propagation delay from OE# | - | - | 150 | 33 | ns |
| G1 | DATA | Driver disable delay from CE1# | - | - | 100 | 7 | ns |
| G1 | DATA | Driver disable delay from OE# | - | - | 100 | 6 | ns |
| G1 | DATA | Driver disable delay from IORD#↑ | - | - | 45 | 22 | ns |
| G1 | DATA | Hold after IORD#↑ | 0 | | - | - | ns |
| G1 | WAIT# | Propagation delay from A7:0 | - | - | 105 | - | ns |
| G1 | WAIT# | Propagation delay from IORD# | - | - | 35 | 31 | ns |
| G1 | WAIT# | Propagation delay from IOWR# | - | - | 35 | 31 | ns |
| G1 | WAIT# | Propagation delay from CE1# | - | - | 40 | 17 | ns |
| G1 | WAIT# | Propagation delay from REG# | - | - | 40 | 17 | ns |
| G1 | INPACK# | Propagation delay from A7:0 | - | - | 115 | 30 | ns |
| G1 | INPACK# | Propagation delay from IORD# | - | - | 45 | 32 | ns |
| G1 | INPACK# | Propagation delay from CE1# | - | - | 50 | 19 | ns |
| G1 | INPACK# | Propagation delay from | - | - | 50 | 34 | ns |

| Type | Signal | Timing | Req. min | Act. min | Req. max | Act. max | Unit |
|---|---|---|---|---|---|---|---|
| | | REG# | | | | | |

The following timings will be conformed to as long as the master clock remains within specified limits:

| Type | Signal | Timing | Min | Max | Unit |
|---|---|---|---|---|---|
| G2 | IREQ# | Delay for assertion after RESET↓ | - | 10 | µs |
| G2 | IREQ# | Delay for deassertion after RESET↓ | - | 5 | s |
| G2 | IREQ# | Pulse width in pulsed-interrupt mode | 500 | - | ns |
| G2 | WAIT# | Low time | - | 3 | µs |

**Timings of EEPROM signals during initialisation**

| Type | Signal | Timing | Min | Nom | Max | Unit |
|---|---|---|---|---|---|---|
| G2 | EECK | Cycle period | 4000 | 72 x $T_{XTALI}$ | - | ns |
| G2 | EECK | Low time | 2000 | 36 x $T_{XTALI}$ | | ns |
| G2 | EECK | High time | 2000 | 36 x $T_{XTALI}$ | | ns |
| G2 | EECS | Set-up before SK↑ | 2000 | 36 x $T_{XTALI}$ | - | ns |
| G2 | EECS | Hold after SK↓ | 2000 | 36 x $T_{XTALI}$ | - | ns |
| G2 | EECS | CS low time | 2000 | 36 x $T_{XTALI}$ | | ns |
| G2 | SBDIO | Set-up before SK↑ | 2000 | 36 x $T_{XTALI}$ | - | ns |
| G2 | SBDIO | Hold after SK↑ | 2000 | 36 x $T_{XTALI}$ | - | ns |
| A2 | SBDIO | Delay to valid after SK↑ | - | 35 x $T_{XTALI}$ | 2187 | ns |
| A2 | SBDIO | Hold after SK↑ | - | 35 x $T_{XTALI}$ | 2187 | ns |
| A2 | SBDIO | Delay to high-Z after SK↑ | - | 35 x $T_{XTALI}$ | 2187 | ns |

These values are calculated using the worst-case XTALI period of 45 ns. 36 x $T_{XTALI}$ is 2.25 us for a 16 MHz clock.

# 5. Operating conditions

## 5.1. Use of LIVIO

The eight function signals and the three EEPROM signals have their I/Os powered by, and referenced to, the LIVIO pin. All other I/Os use the core voltage, VDD. LIVIO should always be kept within the range 0V to VDD, and should be at least 2.7V for correct logic signalling. It is expected that LIVIO be connected to the Bluecore VDD_PADS in Bluetooth applications, and to VDD in other applications.

The configuration EEPROM should draw its power from the supply connected to LIVIO, as the EEPROM interface pins are part of the low-voltage I/O cluster. In applications where the power from the supply connected to LIVIO is switched on and off for power saving, the designer must ensure that the EEPROM power supply is at a valid level at the end of a reset pulse, at which point the BB16CF950+ will attempt to read data from the serial EEPROM. The BB16CF950+ will drive the PWRON pin high during the reset pulse and the reading of the EEPROM, but when the reset pulse is short it may be necessary to use a monostable circuit between the host connector RESET pin and the RESET input on the BB16CF950+, to allow time for the supply voltage to ramp up.

## 5.2. Decoupling capacitors

Place a 1nF and a 22nF decoupling capacitor as close as possible to each VDD and LIVIO pin. The 1nF should be the closer of the two.

Place a decoupling capacitor of between 1 and 5µF near to the BB16CF950+.

Tracks between decoupling capacitors, the BB16CF950+, and power planes should be short and wide for minimum impedance.

## 5.3. Absolute maximum ratings

| | Min | Max | Unit |
|---|---|---|---|
| Supply voltage, VDD or LIVIO | -0.5 | 7.0 | V |
| Input voltage | -0.5 | VDD+7.0 | V |
| Storage temperature | -65 | 150 | ºC |

## 5.4. Recommended operating conditions

| | Min | Nom | Max | Unit |
|---|---|---|---|---|
| Supply voltage, VDD | 3.13 | 3.3 or 5.0 | 5.5 | V |
| Secondary I/O voltage, LIVIO | 2.7 | 3, 5 or 0 | VDD | V |
| Operating free-air temperature, $T_A$ | 0 | 25 | 70 | ºC |
| Oscillator/clock speed | 14 | | 22.2 | MHz |

## 5.5. I/O Electrical specifications

| Name | Pin | Input type | Output type | Output drive | Supply $V_{SUPP}$ |
|---|---|---|---|---|---|
| A0 | 1 | I1 | - | - | VDD |
| A1 | 2 | I1 | - | - | VDD |
| A2 | 4 | I1 | - | - | VDD |
| A3 | 5 | I1 | - | - | VDD |
| A4 | 41 | I1 | - | - | VDD |
| A5 | 42 | I1 | - | - | VDD |
| A6 | 43 | I1 | - | - | VDD |
| A7 | 44 | I1 | - | - | VDD |
| D0 | 48 | I1 | O1 | 6mA | VDD |
| D1 | 46 | I1 | O1 | 6mA | VDD |
| D2 | 45 | I1 | O1 | 6mA | VDD |
| D3 | 19 | I1 | O1 | 6mA | VDD |
| D4 | 18 | I1 | O1 | 6mA | VDD |
| D5 | 17 | I1 | O1 | 6mA | VDD |
| D6 | 14 | I1 | O1 | 6mA | VDD |
| D7 | 13 | I1 | O1 | 6mA | VDD |
| RESET | 6 | I1 | - | - | VDD |
| CE1# | 15 | I1-PU | - | - | VDD |
| REG# | 3 | I1-PU | - | - | VDD |
| OE# | 11 | I1 | - | - | VDD |
| WE# | 8 | I1 | - | - | VDD |
| IORD# | 10 | I1-PU-S | - | - | VDD |
| IOWR# | 9 | I1-PU-S | - | - | VDD |
| IREQ# | 7 | - | O1 | 3mA | VDD |
| READY | | | | | |
| WAIT# | 39 | - | O1 | 3mA | VDD |
| STSCHG# | 40 | - | O1 | 3mA | VDD |
| INPACK# | 20 | - | O1 | 3mA | VDD |
| USER0 | 29 | I1 | O1 | 3mA | LIVIO |
| EECS | 27 | - | O1 | 3mA | LIVIO |
| EECK | 26 | - | O1 | 3mA | LIVIO |
| SBDIO | 28 | I1 | O1 | 3mA | LIVIO |
| LI0 | 31 | - | O1 | 3mA | LIVIO |
| LI1 | 30 | I1 | - | - | LIVIO |
| LI2 | 32 | - | O1 | 3mA | LIVIO |
| LI3 | 34 | I1 | - | - | LIVIO |
| LI4 | 35 | I1 | - | - | LIVIO |
| LI5 | 33 | - | O1 | 3mA | LIVIO |
| LI6 | 38 | - | O1 | 3mA | VDD |

| Name | Pin | Input type | Output type | Output drive | Supply $V_{SUPP}$ |
|---|---|---|---|---|---|
| LI7 | 36 | - | O1 | 3mA | LIVIO |
| XTALI | 23 | I2 | - | - | VDD |
| XTALO | 22 | I2 | O2 | | VDD |

A –PU suffix on an input type description designates an internal pull-up resistor.

A –S suffix on an input type description designates a Schmitt trigger input.

In the tables below, $V_{SUPP}$ refers to the I/O cell supply voltage of either VDD or LIVIO, as shown in the table above.

### i). DC Characteristics: input type I1, supply at +5V ± 10%

| Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|
| Input low voltage, VIL | 0.0 | | 0.8 | V |
| Input high voltage, VIH | 2.2 | | $V_{SUPP}$ | V |
| Schmitt trigger positive threshold, VT+ | | | 1.5 | V |
| Schmitt trigger negative threshold, VT+ | 1.0 | | | V |
| Input leakage (no pull-up) | | ±1 | ±5 | µA |
| Input leakage (with internal pull-up) | | -50 | | µA |

### ii). DC Characteristics: input type I1, supply at +2.7 - 3.63V

| Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|
| Input low voltage, VIL | 0.0 | | 0.8 | V |
| Input high voltage, VIH | 2.0 | | $V_{SUPP}$ | V |
| Schmitt trigger positive threshold, VT+ | | | 1.5 | V |
| Schmitt trigger negative threshold, VT+ | 0.9 | | | V |
| Input leakage (no pull-up) | | | ±1 | µA |
| Input leakage (with internal pull-up) | | -20 | | µA |

### iii). DC Characteristics: input type I2, supply at +2.7 - 3.63V or +5V ± 10%

| Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|
| Input low voltage, VIL | 0.0 | | 0.3 $V_{SUPP}$ | V |
| Input high voltage, VIH | 0.7 $V_{SUPP}$ | | $V_{SUPP}$ | V |
| Input leakage (no pull-up) | | ±1 | ±5 | µA |

### iv). DC Characteristics: output type O1, supply at +5V ± 10%

| Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|
| Output low voltage, VOL (sinking rated current) | | | 0.4 | V |
| Output high voltage, VOH (sourcing rated current) | 2.4 | | | V |
| 3-state output leakage current, IOZ | | ±1 | ±5 | µA |

### v). DC Characteristics: output type O1, supply at +2.7 - 3.63V

| Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|
| Output low voltage, VOL (sinking 50% of rated current) | | | 0.4 | V |
| Output high voltage, VOH (sourcing 33% of rated current) | 2.4 | | | V |
| 3-state output leakage current, IOZ | | | ±1 | µA |

### vi). DC Characteristics: output type O2, supply at +5V ± 10%

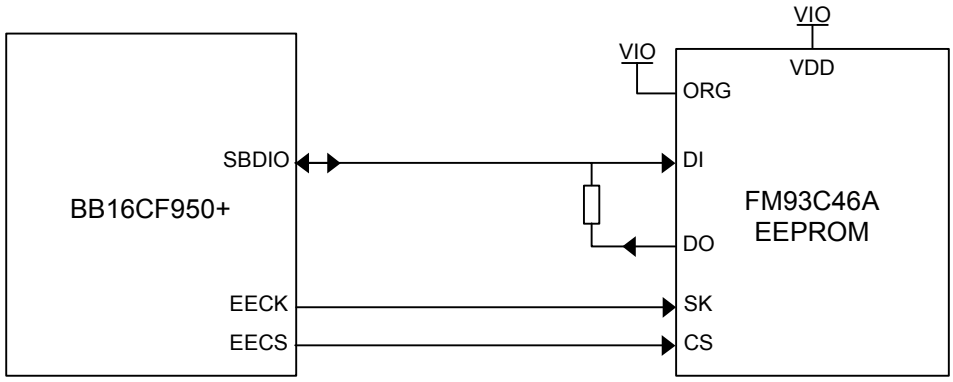
| Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|
| Output low voltage, VOL (sinking rated current) | | | 0.4 | V |
| Output high voltage, VOH (sourcing rated current) | 3.9 | | | V |

| | | | | |
|---|---|---|---|---|
| 3-state output leakage current, IOZ | | ±1 | ±5 | µA |

# 6. Initialisation

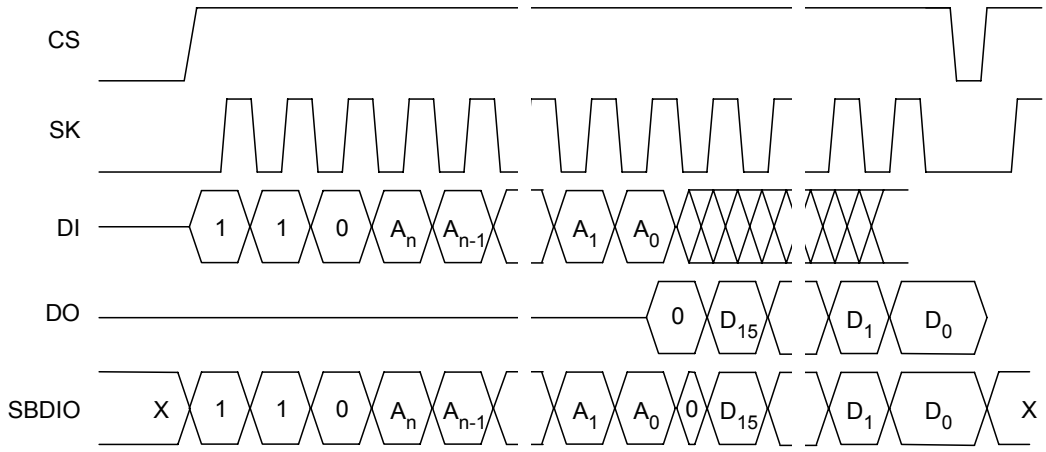
After the host bus RESET signal has been asserted and then released, the BB16CF950+ drives the IREQ# output low to signal that the card is not ready for access (until the card has been configured for I/O accesses, the IREQ# signal acts as a READY signal to the host). The BB16CF950+ then reads the attached serial EEPROM to obtain configuration and CIS information. The EEPROM must be in 16-bit mode, and connected for 3-wire operation (see diagram below).

Because the SBDIO pin is also used for Bluetooth SPI access, the BB16CF950+ EEPROM pins are operated at the LIVIO voltage level, and the EEPROM must also be powered at this voltage.




**Example circuit for EEPROM connection**

The BB16CF950+ performs repeated read operations as shown below: it clocks in a command sequence of "110" followed by 6-9 address bits, and then expects to see a 16-bit data value driven on the SBDIO line.




After the deassertion of RESET, the first data word in the EEPROM is read, and the result is used to determine the number of address bits which the EEPROM uses. In order for this to work, the high byte of the first data word must contain 10 hex.

## 6.1. EEPROM data format

The first two data words in the EEPROM contain a synchronisation byte and three bytes of configuration data. The synchronisation byte allows EEPROMs of different sizes to be used. The configuration data allows the reset value of the MFR, UCR, BPR, USR0R and USR1R registers to be defined.

Following bytes from the EEPROM are read into the CIS RAM block, from where they may later be read by the host using attribute memory reads.

The structure of the data in the EEPROM is given by the table below:

| EEPROM address | High byte (D15:8) | Low byte (D7:0) |
|---|---|---|
| 00 | Synchronisation byte: 10 hex | Configuration byte 1 |
| 01 | Configuration byte 2 | Configuration byte 3 |
| 02 | CIS data for address 02 | CIS data for address 00 |
| 03 | CIS data for address 06 | CIS data for address 04 |
| … | … | … |
| N | CIS data for address 4n-6 | CIS data for address 4n-8 |

For EEPROMs of less than 128 bytes capacity, the EEPROM may be read repeatedly until 112 bytes have been written to the CIS RAM. The resulting garbage data in the CIS RAM will be irrelevant because it will not form part of the CIS linked list of tuples.

The data for the synchronisation byte and the register bits initialised by the configuration bytes are detailed below:

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Sync. Byte | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Config 1 | MFR7 | MFR6 | UCR3 | UCR2 | UCR1 | UCR0 | MFR1 | MFR0 |
| Config 2 | TEST | BPR4 | BPR3 | BPR2 | BPR1 | SFR4 | SFR3 | SFR2 |
| Config 3 | ERSC | USR1R2 | USR1R1 | USR1R0 | RFU | USR0R2 | USR0R1 | USR0R0 |

The configuration bytes from the serial EEPROM configure the reset value of these register bits. The register values may be changed by host I/O accesses, but the registers will revert to the configured settings when a soft reset of the UART is performed. A hard reset (assertion of the RESET pin) will cause the EEPROM to be read again and the registers reset from the EEPROM data.

Bit 7 of configuration byte 2 does not initialise any register, but if it is set then the initialisation process stops after reading the configuration bytes, i.e. no data is read from the EEPROM to initialise the CIS. This bit is provided to increase the speed of chip test and simulation, and should not be set for production hardware.

Bit 7 of configuration byte 3 sets whether the extended registers safety catch (ERSC) is enabled. If this bit is set, then after the UART is reset, writes to the IRSR will be ignored and the extended registers will be inaccessible. See section 9.31 for details.

The table below provides a summary of the usage of the configuration bits: refer to the relevant register description for details.

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| Sync. Byte | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Config 1 | Enable ESR | Wake on interrupt | Enable low power | Extra THRE interrupts | Deep FIFOs | Force AFC on | Mode(1) | Mode(0) |
| Config 2 | Don't read CIS from EEPROM | Auto-reset after power up | Power down when CSR4 set | Power down when CSR2 set | Power down when function disabled | SF=16 | SF=8 | SF=4 |
| Config 3 | Extended regs safety catch | USER1 Output clock | USER1 Set state | USER1 Output enable | RFU | USER0 Invert on reset | USER0 Set state | USER0 Output enable |

## 6.2. Pin states before and during reset

Between the power-up of the device and the assertion of the host RESET signal, the value of any pin which may be an output will be indeterminate: they may be driving high, driving low, or some may be tristated. Some protection (e.g. a low-value series resistor) may be required if any of USER0, pin 36, or pin 38 are to be used as an input.

Between the assertion of the host RESET signal and the completion of the reading of the EEPROM, all EEPROM-configured register bits will be cleared to 0, and pin 38 will be driven high to ensure that the EEPROM is powered in Bluetooth mode.

# 7. CompactFlash/PC Card interface

**Acknowledgement:** the host bus signal descriptions and the definitions of the COR, CSR and PRR registers are based on text from the CompactFlash Specification Revision 1.4 published by the CompactFlash Association, reproduced here by permission.

The BB16CF950+ is always in one of two states: memory only, or enabled for I/O access (function enabled). Attribute memory reads and writes are allowed in either state. Common memory accesses are not supported in either state.

## 7.1. Host bus signal descriptions

A7:0 Address bus. These address lines along with the REG# signal are used to select an I/O port address registers within the card, or a byte in the card's information structure and its configuration control and status registers.

D7:0 Data bus. These lines carry the Data, Commands and Status information between the host and the card. D0 is the LSB.

CE1# Card Enable 1. The CE2# and CE1# lines input signals are used both to select the card and to indicate to the card whether a byte or a word operation is being performed. CE2# always accesses the odd byte of the word. CE1# accesses the even byte or the odd byte of the word depending on A0 and CE2#. The BB16CF950+ only requires connection to CE1# because it only supports 8-bit accesses.

OE# Output Enable. This signal is used to read the CIS and configuration registers.

WE# Write Enable. This signal is used for writing the configuration registers.

READY, IREQ# Ready. In Memory Mode this signal is set high when the Card is ready to accept a new data transfer operation and held low when the card is busy. The Host memory card socket must provide a pull-up resistor.

At power up and at Reset, the READY signal is held low (busy) until the card has completed its power up or reset function. No access of any type should be made to the card during this time.

I/O Operation - After the card has been configured for I/O operation, this signal is used as Interrupt Request. This line is strobed low to generate a pulse mode interrupt or held low for a level mode interrupt. The BB16CF950+ supports both of these modes, selected by the host using the function control registers.

REG# This signal is used during Memory Cycles to distinguish between Common Memory and Register (Attribute) Memory accesses. High for Common Memory, Low for Attribute Memory.

The signal must also be active (low) during I/O Cycles when the I/O address is on the Bus.

STSCHG# Status Changed. This signal is asserted low to alert the host to changes in the READY and Write Protect states, while the I/O interface is configured. Its use is controlled by the card Configuration and Status Register.

When the card is not configured for I/O, this signal is replaced by BVD1, and remains in a high state (no internal battery).

IORD# I/O Read. This is an I/O Read strobe generated by the host. This signal gates I/O data onto the bus from the card when the card is configured to use the I/O interface.

IOWR# I/O Write. The I/O Write strobe pulse is used to clock I/O data on the Card Data bus into the card controller registers when the card is configured to use the I/O interface. The clocking will occur on the negative to positive edge of the signal (trailing edge).

RESET Card Reset. When the pin is high, this signal Resets the card. The card is Reset only at power up if this pin is left high or open from power-up. The card is also Reset when the Soft Reset bit in the Card Configuration Option Register is set.

WAIT# Extend Bus Cycle. The WAIT# signal is driven low by the card to signal the host to delay completion of a memory or I/O cycle that is in progress.

INPACK# Input Acknowledge. The Input Acknowledge signal is asserted by the card when the card is selected and responding to an I/O read cycle at the address that is on the address bus. This signal is used by the host to control the enable of any input data buffers between the card and the host.

## 7.2. Address map for attribute memory

Accesses to attribute memory are only supported if they are 8-bit accesses to even addresses (CE2#=1 and A0=0). The base address for the Function Control Registers is E0 hex.

| Address (hex) | Offset (decimal) from FCR base | Function |
|---|---|---|
| 00-DE | | CIS data, initialised from EPROM. Writes to this region are not legal and will be ignored. |
| E0 | +0 | Configuration option register (COR) |
| E2 | +2 | Card status register (CSR) |
| E4 | +4 | Pin replacement register (PRR) |
| E8 | +8 | ESR (not in CompactFlash specification) |
| Other addresses | | None: reads return 0, writes are ignored. |

The Socket and Copy Register, the IO Base and Limit Registers, and the Power Management Registers are optional and are not implemented.

For the purposes of CompactFlash and PC Card compliance, the BB16CF950+ is a single-function device, the function comprising the function control registers and the embedded UART, or the external devices when used in local bus mode.

## 7.3. Configuration Option Register (COR)

The Configuration Option Register is used to configure the cards interface, address decoding and interrupt and to issue a soft reset to the card.

The Configuration Option register is organized as follows:

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| SRESET | LevIREQ | Function Configuration Index | | | | | |

| Field | Description |
|---|---|
| SRESET | Soft Reset: setting this bit to one (1), waiting the minimum reset width time and returning to zero (0) places the card in the Reset state. Setting this bit to one (1) is equivalent to assertion of the RESET signal except that the SRESET bit is not cleared. Returning this bit to zero (0) leaves the card in the same un-configured, Reset state as following power-up and hardware reset. This bit is set to zero (0) by |

| | |
|---|---|
| | power-up and hardware reset. |
| LevIREQ | This bit is set to one (1) when Level Mode Interrupt is selected, and zero (0) when Pulse Mode is selected. Set to zero (0) by Reset. |
| Function Configuration Index | Configuration Index: set to zero (0) by reset. It is used to select operation mode of the card as shown below.<br>The effect of possible settings of this field is shown in the table below. |

**Function Configuration Index**

| Value (binary) | Description |
|---|---|
| 0 0 0 0 0 0 | The function is memory mapped, I/O cycles are ignored. |
| 0 0 0 0 0 1 | The function is enabled for I/O access, in normal operation mode. |
| 1 0 0 0 0 1 | The function is placed in a state where I/O access to the internal registers is possible. This setting should only be used in a manufacturing environment, where it ensures that the EEPROM can always be reprogrammed, even when the data on it selects the local bus mode. |
| others | RFU |

## 7.4. Configuration and Status Register (CSR)

The Card Configuration and Status Register contains information about the Card's condition.

The Configuration and Status Register is organized as follows:

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Changed | SigChg | IOIs8 | XE# | Audio | PwrDwn | Intr | IntrAck |

| Field | Type | Description |
|---|---|---|
| Changed | RO | Indicates that one or both of the Pin Replacement register CRdy, or CWProt bits are set to one (1). When the Changed bit is set, -STSCHG Pin 46 is held low if the SigChg bit is a One (1) and the card is configured for the I/O interface. |
| SigChg | R/W | this bit is set and reset by the host to enable and disable a state-change "signal" from the Status Register, the Changed bit control pin 46 the Changed Status signal. If no state change signal is desired, this bit should be set to zero (0) and the STSCHG# signal will be held high while the card is configured for I/O. |
| IOIs8 | R/W | The host sets this bit to a one (1) if the card is to be configured in an 8 bit I/O Mode. This bit is ignored, as the BB16CF950+ can only use an 8-bit data path. |
| XE# | R/W | This bit is set and reset by the host to disable and enable Power Level 1 commands in CF+ cards. If the value is 0, Power Level 1 commands are enabled; if it is 1, Power Level 1 commands are disabled. Default value at power on or after reset is 0. The host may read the value of this bit to determine whether Power Level 1 commands are currently enabled.<br>Power levels are only defined for CF-ATA cards, but the BB16CF950+ can be configured to deassert the PWRON pin when this bit is set. |
| Audio | R/W | This bit is set and reset by the host to enable and disable audio information on SPKR# when the card is configured. |
| PwrDwn | R/W | This bit indicates whether the host requests the card to be in the power saving or active mode. The BB16CF950+ can be configured to deassert the PWRON pin when this bit is set. |
| Intr | RO | Interrupt Request/Acknowledge - This bit represents the internal state of the interrupt request. This value is available whether or not I/O interface has been configured. This signal remains true until the condition that caused the interrupt request has been serviced. |
| IntrAck | RO | Interrupt Acknowledge – Since the BB16CF950+ is a single function device, it ignores this field on writes and always returns zero (0). |

## 7.5. Pin Replacement Register (PRR)

Some pins on the CompactFlash / PC Card interface are reassigned to different uses when the card is enabled for I/O. The Pin Replacement register allows access to these lost signals and detection of changes on them.

The register may be read and written, however, when written the lower 4 bits act as mask bits for changing the corresponding bit of the upper 4 bits.

The upper 4 bits are set when the corresponding bit in the lower 4 bits changes state.

The Pin Replacement Register is organized as follows:

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| CBVD1 | CBVD2 | CREADY | CWProt | RBVD1 | RBVD2 | RREADY | RWProt |

| Field | Description |
|---|---|
| CBVD1 | This bit is set (1) when the corresponding bit, RBVD1, changes state. This bit may also be written by the host. |
| CBVD2 | This bit is set (1) when the corresponding bit, RBVD2, changes state. This bit may also be written by the host. |
| CREADY | This bit is set to one (1) when the bit RREADY changes state. This bit can also be written by the host. |
| CWProt | This bit is set to one (1) when the RWprot changes state. This bit may also be written by the host. |
| RBVD1 | When read, this bit is always one (1).<br>When this bit is written as 1 the corresponding CBVD1 bit is also written. When this bit is written as 0, the CBVD1 bit is unaffected. |
| RBVD2 | When read, this bit is always one (1).<br>When this bit is written as 1 the corresponding CBVD2 bit is also written. When this bit is written as 0, the CBVD2 bit is unaffected. |
| RREADY | This bit is used to determine the internal state of the READY signal. This bit may be used to determine the state of READY as this pin has been reallocated for use as Interrupt Request on an I/O card. When written, this bit acts as a mask for writing the corresponding bit CRREADY. |
| RWProt | This bit is always zero (0) since the card does not have a Write Protect switch. When written, this bit acts as a mask for writing the corresponding bit CWProt. |

## 7.6. Extended Status Register (ESR)

This optional register is located at offset 08H, and contains information about the changes in the card's status. The Extended Status register is organized as follows:

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Event3 | Event2 | Event1 | Req Attn | Enable3 | Enable2 | Enable1 | Req Attn Enable |

| Field | Description |
|---|---|
| Event3 | Reserved for future expansion/definition, is always reset (0) |
| Event2 | Reserved for future expansion/definition, is always reset (0) |
| Event1 | Reserved for future expansion/definition, is always reset (0) |

| | |
|---|---|
| Req Attn | This bit is latched within one(1) ms of an event occurring. An event is a falling edge on RI# when the BB16CF950+ is in RS232 mode, or any enabled UART interrupt when “wake on interrupt” is enabled in the EEPROM.<br>When this bit is set to a one (1), and the Req Attn Enable bit is set to a one (1), the Changed bit in the Configuration and Status register will also be set to a one (1), and if the SigChg bit in the Configuration and Status register has also been set by the host, then the STSCHG# pin will be asserted. The host writing a one (1) to this bit will reset it to zero (0). Writing a zero (0) to this bit will not have any effect. |
| Enable3 | Reserved for future expansion/definition, is always reset (0) |
| Enable2 | Reserved for future expansion/definition, is always reset (0) |
| Enable1 | Reserved for future expansion/definition, is always reset (0) |
| Req Attn Enable | Setting this bit to a one (1) enables the setting of the Changed bit in the Configuration and Status register when the Req Attn bit is set. When this bit is reset to a zero (0), this feature is disabled. The state of the Req Attn bit is not affected by the Req Attn Enable bit. |

The register may be read or written. The upper 4 bits are latched to a one (1) when the corresponding event occurs on the PC Card (for example in the case of the Req Attn bit, when ringing occurs on the phone line on a modem card). When one of these upper four bits is latched and the corresponding enable bit in the lower nibble is also set, the Changed bit in the Configuration and Status register will be set, and if the SigChg bit in the Configuration and Status register is also set, (and the card is configured for I/O mode) then the STSCHG# pin will be asserted. The host writing a one (1) to one of the upper 4 bits will clear that bit. Writing a zero to one of the upper 4 bits will have no effect.

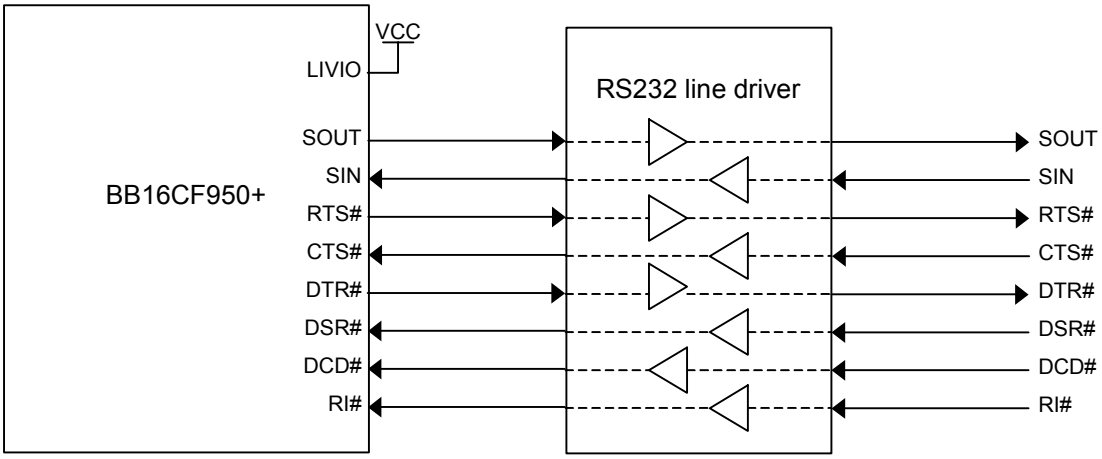
The lower four bits will return their current state when they are read. All bits of this register are cleared to zero (0) by RESET or SRESET.

Setting one of the lower 4 bits enables the corresponding upper bit to be OR'ed into the Changed bit in the Configuration and Status register. When these lower bits are cleared, the setting of the Changed bit is disabled for the corresponding event. The state of the upper bits is not affected by the value written to the lower bits.

# 8. Local interface modes

## 8.1. RS232 mode

This is the mode which is closest to the behaviour of a TL16C750. The local interface provides a full set of RS232 handshaking lines, which may be connected to an RS232 line driver/receiver as shown below.



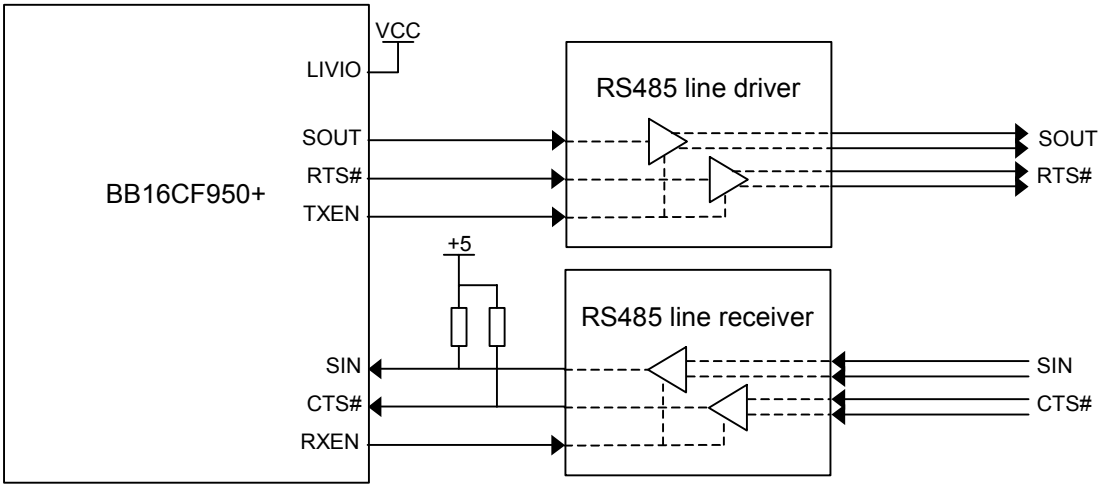

**Example circuit for RS232 mode**

## 8.2. RS485 mode

In RS485 mode, the DTR, DSR, DCD and RI signals are not connected to any pins. The DCD and DSR signals are internally driven to an asserted state, and the RI input is driven to an asserted state. (MSR7:5 will always read as 101b, and MSR3:1 as 000b.)

Two of the local interface pins become “transmitter enable” and “receiver enable” signals, allowing a design to support both full duplex and half duplex operation, with either autogating or direct software control of the direction of data flow.

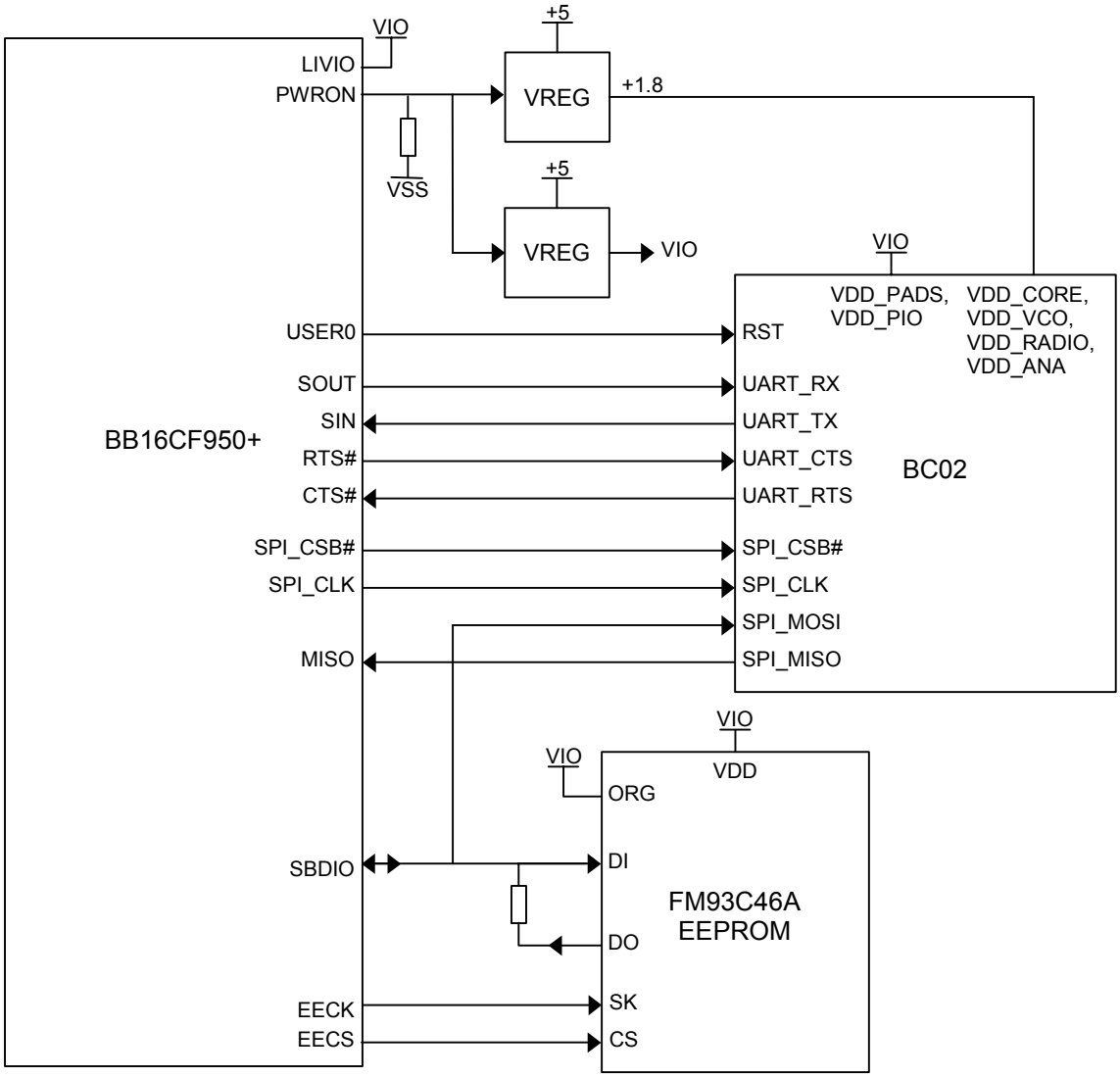



**Example circuit for RS485 mode**

## 8.3. Bluetooth mode

The Bluetooth mode provides access to the SIN/SOUT data signals, and the RTS/CTS handshaking pair of the UART, and uses the other configurable pins to provide a power control signal (PWRON) and an interface to an SPI bus.

As in RS485 mode, the DTR, DSR, DCD and RI signals are not connected to any pins. The DCD and DSR signals are internally driven to an asserted state, and the RI input is driven to a deasserted state. (MSR7:5 will always read as 101b, and MSR3:1 as 000b.)




**Example circuit for Bluetooth mode**

All regulators for the Bluecore must have the same enable signal, and the rail used for VDD_PADS should be fed into LIVIO and should also be used to power the serial EEPROM (see section 5.1). A pull-down resistor is recommended on the PWRON signal because the BB16CF950+ may leave it as an input between card power-up and reset.

## 8.4. Local bus mode

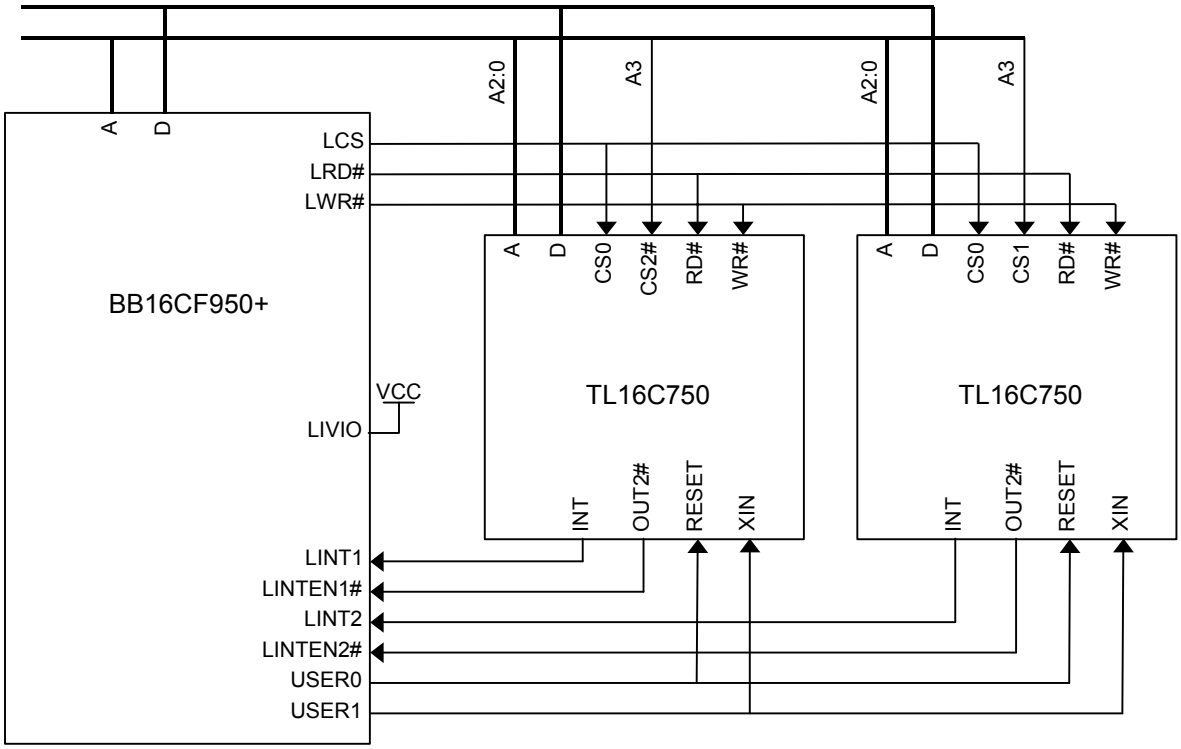
In local bus mode, the I/O registers on the BB16CF950+ are not accessible. The BB16CF950+ just provides the CIS and FCRs in attribute space, and gives local chip select, read and write strobes which are asserted for I/O accesses. External devices with a suitable "address, data, chip select, read and write strobe" interface can then be connected to these signals and to the host address and data busses.

With the connection of two external UARTs in mind, this mode provides two active-high interrupt inputs and two active-low interrupt enable inputs. The BB16CF950+ will generate a host bus interrupt when

(interrupts are enabled) and ( (LINT1 and not LINTEN1#) or (LINT2 and not LINTEN2#) ).

Pulsed or level-mode interrupts will be generated by the BB16CF950+ as requested by the host.




**Example circuit for 2-port serial card using local bus mode**

The timings and signal loads will need to be checked for any given design.

# 9. UART function

## 9.1. Programming

To prepare the UART for communication, it is necessary to first configure the serial channel using the control registers LCR, SFR, DLL and DLM. These set the number of data and stop bits, the parity setting and the baud rate. These registers can be changed at any time, but if data is being received or transmitted then corruption of the serial data is likely to occur.

It is also a good idea to enable FIFOs using FCR and UCR, to decrease the number of data-transfer services the UART will require. The trigger levels can also set at this stage using RFTR, RITR and TITR, although the TL16C550-compatible method using FCR7:6 will still work. If wanted, auto-flow control should be enabled by writing the MCR, and the same register sets the initial state of the output handshaking lines.

Once the serial channel is configured, interrupts can be enabled by writing IER and setting MCR3.

The interrupt handler can read the IIR to determine what type of event needs servicing: the interrupt types are prioritised so that if more than one event needs servicing, the most urgent one is indicated.

A “transmitter FIFO empty” interrupt is cleared as soon as the IIR is read, so if there is no data waiting to be transmitted then no further action is needed. To restart the flow of transmitted data, the usual practice is for the user-mode part of the device driver to add the data to the software transmit queue and then “kickstart” transmission by re-writing to the IER with its current value (with bit 0 set). This will re-enable the “transmitter FIFO empty” interrupt and the interrupt handler will handle the transfer of transmit data to the UART, pushing another block every time the FIFO becomes empty.

## 9.2. Accessible registers

The internal registers of the UART are listed below, organised by function and showing the full name and mnemonic.

| REGISTER SELECTION | MNEMONIC |
|---|---|
| Indexed register select register | IRSR |
| Line control register (bit 7) | LCR |

| UART DATA | MNEMONIC |
|---|---|
| Receiver buffer register | RBR |
| Transmitter holding register | THR |

| UART CONTROL | MNEMONIC |
|---|---|
| LSB divisor latch | DLL |
| MSB divisor latch | DLM |
| Interrupt enable register | IER |
| FIFO control register | FCR |
| Line control register | LCR |
| Modem control register | MCR |
| Synchronisation factor register | SFR |
| UART configuration register | UCR |
| Half-duplex control register | HDCR |
| Receive FIFO flow-control trigger register | RFTR |
| Receive FIFO interrupt trigger register | RITR |
| Transmit FIFO interrupt trigger register | TITR |

| UART STATUS (read only) | MNEMONIC |
|---|---|
| Interrupt identification register | IIR |
| Line status register | LSR |
| Modem status register | MSR |
| Chip ID register | CIDR |
| Receive FIFO level register | RFLR |
| Transmit FIFO level register | TFLR |

| CHIP CONTROL | MNEMONIC |
|---|---|
| USER0 control register | USR0R |
| USER1 control register | USR1R |
| Mode and features register | MFR |
| Bluetooth power control register | BPR |

| BUILD AND TEST | MNEMONIC |
|---|---|
| Scratch pad register | SCR |
| Inverted scratch pad register | ISCR |
| Microwire and SPI control register | SBR |

Individual bits within the registers are referred to by the register mnemonic with the bit number appended. For example, LCR7 refers to bit 7 of the line control register.

The register accessed when an I/O read or write is performed will depend on the address lines A2:0, the divisor latch access bit (DLAB, which is LCR7), and the Indexed Register Select register (IRSR).

The transmitter holding register and receiver buffer register are used to transfer data for transmission and received data respectively. These are eight-bit registers, but the serial data may be 5, 6, 7 or 8 bits long: data is right-justified and padded with zeroes on the left. The UART always receives and transmits bit 0 first. The THR and RBR can be accessed at the same time as serial data transmission and reception are taking place, because the serialiser and deserialiser are separate from the data buffers/FIFOs.

**Register Selection**

| IRSR | DLAB | A2:0 | Mnemonic | Register |
|---|---|---|---|---|
| X | 0 | 0 | RBR | Receiver buffer register (read only) |
| X | 0 | 0 | THR | Transmitter holding register (write only) |
| X | 1 | 0 | DLL | LSB divisor latch |
| X | 1 | 1 | DLM | MSB divisor latch |
| X | 0 | 1 | IER | Interrupt enable register |
| X | X | 2 | IIR | Interrupt identification register (read only) |
| X | X | 2 | FCR | FIFO control register (write only) |
| X | X | 3 | LCR | Line control register |
| X | X | 4 | MCR | Modem control register |
| X | X | 5 | LSR | Line status register (read only) |
| X | X | 6 | MSR | Modem status register (read only) |
| X | X | 6 | IRSR | Indexed register select register (write only) |
| 0 | X | 7 | SCR | Scratch pad register |
| 1 | X | 7 | ISCR | Inverted scratch pad register |
| 2 | X | 7 | CIDR | Chip ID register (read only) |
| 3 | X | 7 | SFR | Synchronisation factor register |
| 4 | X | 7 | USR0R | USER0 control register |
| 5 | X | 7 | USR1R | USER1 control register |
| 6 | X | 7 | UCR | UART configuration register |
| 7 | X | 7 | MFR | Mode and features register |
| 8 | X | 7 | RFLR | Receive FIFO level register |
| 9 | X | 7 | TFLR | Transmit FIFO level register |
| 10 | X | 7 | HDCR | Half-duplex control register |
| 11 | X | 7 | SBR | Microwire and SPI control register |
| 12 | X | 7 | BPR | Bluetooth power control register |
| 13 | X | 7 | RFTR | Receive FIFO flow-control trigger register |
| 14 | X | 7 | RITR | Receive FIFO interrupt trigger register |
| 15 | X | 7 | TITR | Transmit FIFO interrupt trigger register |
| >15 | X | 7 | - | Reserved for future use – do not read or write |

X = irrelevant, 0 = low level, 1 = high level

The system programmer, using the host, can access any of the UART registers, as summarised in the table below.

## Summary of Accessible Registers in the BB16CF950+

| REGISTER MNEMONIC, ACCESS† | REGISTER BIT NUMBER | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **BIT 7** | **BIT 6** | **BIT 5** | **BIT 4** | **BIT 3** | **BIT 2** | **BIT 1** | **BIT 0** |
| RBR (read only) A=0, DLAB=0 | Data Bit 7 (MSB) | Data Bit 6 | Data Bit 5 | Data Bit 4 | Data Bit 3 | Data Bit 2 | Data Bit 1 | Data Bit 0 (LSB) |
| THR (write only) A=0, DLAB=0 | Data Bit 7 | Data Bit 6 | Data Bit 5 | Data Bit 4 | Data Bit 3 | Data Bit 2 | Data Bit 1 | Data Bit 0 |
| DLL A=0, DLAB=1 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| DLM A=1, DLAB=1 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| IER A=1, DLAB=0 | 0 | 0 | Low power enable (ignored) | Sleep mode enable (ignored) | (EDSSI) Enable modem status interrupt | (ERLSI) Enable receiver line status interrupt | (ETHREI) Enable transmitter buffer empty interrupt | (ERBFI) Enable received data available interrupt |
| FCR (write only) A=2 | Receiver Trigger (MSB) | Receiver Trigger (LSB) | 64 Byte FIFO Enable | 0 | DMA mode select (ignored) | Transmitter FIFO reset | Receiver FIFO reset | FIFO enable |
| IIR (read only) A=2 | FIFOs Enabled‡ | FIFOs Enabled‡ | 64 Byte FIFO Enabled | 0 | Interrupt ID Bit 3‡ | Interrupt ID Bit 2 | Interrupt ID Bit 1 | 0 if interrupt pending |
| LCR A=3 | (DLAB) Divisor latch access bit | Set break | Stick parity | (EPS) Even parity select | (PEN) Parity enable | (STB) Number of stop bits | (WLSB1) Word length select bit 1 | (WLSB0) Word length select bit 0 |
| MCR A=4 | 0 | 0 | AFE | Loop | OUT2 (interrupt enable) | OUT1 | RTS | DTR |
| LSR (read only) A=5 | Error in Receiver FIFO‡ | Transmitter Empty (TEMT) | Transmit holding register (THRE) | Break Interrupt (BI) | Framing Error (FE) | Parity Error (PE) | Overrun Error (OE) | Data Ready (DR) |
| MSR (read only) A=6 | (DCD) Data carrier detect | (RI) Ring indicator | (DSR) Data set ready | (CTS) Clear to send | (ΔDCD) Delta data carrier detect | (TERI) Trailing edge ring indicator | (ΔDSR) Delta data set ready | (ΔCTS) Delta clear to send |
| IRSR (write only) A=6 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| SCR A=7, IRSR=0 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| ISCR A=7, IRSR=1 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| CIDR (read only) A=7, IRSR=2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| SFR A=7, IRSR=3 | RFU | RFU | RFU | SF=16 | SF=8 | SF=4 | RFU | RFU |
| USR0R A=7, IRSR=4 | RFU | RFU | RFU | RFU | Read state | Invert on reset | Set state | Output enable |
| USR1R A=7, IRSR=5 | RFU | RFU | RFU | RFU | Read state | Output clock | Set state | Output enable |
| UCR A=7, IRSR=6 | RFU | RFU | RFU | RFU | Enable low-power mode | Enable extra THRE interrupts | Enable deep FIFOs | Force AFC on |
| MFR A=7, IRSR=7 | Enable ESR | Wake on interrupt | RFU | RFU | RFU | RFU | Mode Bit 1 | Mode Bit 0 |
| RFLR (read only) A=7, IRSR=8 | Error in Receiver FIFO | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| TFLR (read only) A=7, IRSR=9 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| HDCR A=7,IRSR=10 | RFU | RFU | RFU | RFU | Hold CTS | Half-duplex mode bit 2 | Half-duplex mode bit 1 | Half-duplex mode bit 0 |
| SBR A=7, RSR=11 | SPI_CSB | SPI_CLK | RFU | MISO (RO) | EECS | EECK | SBDIO output enable | SBDIO data in/out |
| BPR A=7, RSR=12 | RFU | RFU | Doing auto-reset (RO) | Enable auto-reset | CSR4 power down | CSR2 power down | COR power down | Power down now |
| RFTR A=7, RSR=13 | RFU | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| RITR A=7, RSR=14 | RFU | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| TITR A=7, RSR=15 | RFU | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |

† In this column, 'A' refers to address bits 2:0. ‡ These bits are always 0 when FIFOs are disabled.

## 9.3. Indexed Register Select register (IRSR)

The Indexed Register Select register is a write-only register at the same location as the Modem Status Register. Writing a value to this register selects the register that is accessible at what would be the location of the scratchpad register in a TL16C750. The reset value of the IRSR is 0, which selects the scratchpad register.

## 9.4. Master reset

It is recommended that a hard or soft reset should be performed after power up.

The following tables summarise the effect of reset on the UART circuits:-

### *Effect of RESET on UART signals*

| UART Signal | Reset control | Signal reset state |
|---|---|---|
| DTR# | Reset | High |
| RTS# | Reset | High |
| SOUT | Reset | High |

### *Effect of RESET on UART registers*

| UART Register | Reset control | Register reset state |
|---|---|---|
| Line status register (LSR) | Reset | Bits 7,4,3,2,1,0 are cleared<br>Bits 6 and 5 are set. |
| Modem control register (MCR) | Reset | All bits are cleared<br>Note bits 7–6 are permanently cleared |
| Interrupt enable register (IER) | Reset | All bits are cleared<br>Note bits 7–6 are permanently cleared |
| FIFO control register (FCR) | Reset | All bits are cleared |
| Interrupt identification register (IIR) | Reset | Bits 7,6,3,2,1 are cleared<br>Bit 0 is set<br>Note bit 5 is permanently cleared |
| Line control register (LCR) | Reset | All bits are cleared |
| Transmit FIFO interrupt trigger register (TFTR) | Reset | All bits are cleared |
| Modem status register (MSR) | Reset | Bits 3 –0 are cleared<br>Bits 7 –4 are input signals |
| Indexed register select register (IRSR) | Reset | All bits are cleared |
| Scratch Register (SCR) | Reset | All bits are cleared |
| Divisor Latch (LSB and MSB) Registers | Reset | All bits are cleared |
| Receiver Buffer Registers (RBR) | Reset | All bits are cleared |
| Transmitter Holding Registers (THR) | Reset | All bits are cleared |
| Half-duplex control register | Reset | Bit 3 is cleared. Bits 2-0 are set<br>Note bits 7–4 are permanently cleared. |
| Receive FIFO flow-control trigger register (RFTR) | Reset | All bits are cleared |
| Receive FIFO interrupt trigger register (RITR) | Reset | All bits are cleared |
| Modem Features Register (MFR)<br>UART Control Register (UCR)<br>Bluetooth Power Register (BPR)<br>User 0 Register (USR0R)<br>User 1 Register (USR1R) | Reset | Value defined by data in EEPROM |

### *Effect of RESET on UART Interrupts*

| Interrupt | Reset control | Interrupt reset state |
|---|---|---|
| Interrupt type - modem status changes | Reset/Read MSR | Low |
| Interrupt type - receiver data ready | Reset/Read RBR | Low |
| Interrupt type - RCVR errors | Reset/Read LSR | Low |
| Interrupt type - THRE | Reset/Read IIR/Write THR | Low |

### *Effect of RESET on UART FIFOs*

| FIFO | Reset control | FIFO reset state |
|---|---|---|
| Receiver FIFO | Reset<br>FCR1–FCR0<br>∆FCR0 | FIFO empty |
| Transmitter FIFO | Reset<br>FCR1–FCR0<br>∆FCR0 | FIFO empty |

## 9.5. Serial data format

A logic '0' in the RBR or THR corresponds to a logic low on SIN or SOUT, and a logic '1' in the Receiver Buffer Register (RBR) or Transmitter Holding Register (THR) corresponds to a logic high on SIN or SOUT.

Bit 0 is always the least significant bit (LSB)

The first bit to be serially transmitted or received is Bit 0.

A start bit or 'line break' state corresponds to a logic low on SIN or SOUT.

A stop bit or inter-byte 'marking' state corresponds to a logic high on SIN or SOUT.

## 9.6. Receiver Section

The status of the receiver is given by the Line Status Register (LSR). For more details, refer to Section 9.10.

The control of the receiver section and format of the data characters such as number of data bits, parity, etc is controlled by the Line Control Register (LCR). Note if parity is used (LCR3) then the polarity of parity LCR4 is required. For more details, refer to Section 9.9

As serial asynchronous data is fed into the receiver serial data input terminal SIN, the UART continually looks for a high-to-low transition. Upon detection of the transition, an internal counter is reset and counts the SF× clock input to SF/2, which is the centre of the start bit. (SF is the Synchronisation Factor)

The receiver is prevented from assembling a false data character caused by noise on the SIN input, by verification of the start bits. Note: The start bit is valid only if SIN is still low.

## 9.7. Receiver buffer register (RBR)

The UART receiver section contains a Receiver Buffer Register (RBR) which is a FIFO of selectable depth and a Receiver Deserialiser Register (RDR). Data fed into the receiver serial data input terminal SIN is deserialised by the RDR and is fed into the RBR.

The control of the receiver section and format of the data characters such as number of data bits, parity, etc is controlled by the Line Control Register (LCR). Note if parity is used (LCR3) then the polarity of parity LCR4 is required. For more details, refer to Section 9.9

The receiver timing is supplied by the SF (Synchronisation Factor) × receiver clock (RCLK).

In FIFO modes, the FIFO control register (FCR) is used to enable and reset the receiver FIFO and also can be used to set data trigger levels for when interrupts are generated. For more details see Section 9.13.

In non FIFO mode (16C450), when the received data available interrupt is enabled, an interrupt is generated when a character is placed in the receiver buffer register. When the RBR is read, the interrupt is cleared.

## 9.8. Transmitter holding register and multiplexer register (THR and TMR)

The UART transmitter section contains a Transmitter Holding Register (THR), which is a FIFO of selectable depth and a transmitter multiplexer register (TMR). The THR receives data off the internal data bus and moves it into the TMR, while the transmitter is idle, which serializes the data and outputs it to the transmitter data serial output terminal SOUT.

The transmitter timing is supplied by the baud clock generator.

The control of the receiver section and format of the data characters such as number of data bits, parity, etc is controlled by the Line Control Register (LCR). Note if parity is used (LCR3) then the polarity of parity LCR4 is required. For more details, refer to Section 9.9

In FIFO modes, the FIFO control register (FCR) is used to enable and reset the transmitter FIFOs and also can be used to set data trigger levels for when interrupts are generated. For more details see Section 9.13.

In non FIFO mode (16C450), when the transmitter holding register empty interrupt is enabled, an interrupt is generated when the THR is empty. When a character is loaded into the register, the interrupt is cleared.

For RS485 mode, during the output of start, data, parity or stop bits, the TXEN output is asserted (high). When the transmitter is idle and an inter-character 'marking' state is being output, the TXEN output is de-asserted (low).

## 9.9. Line control register (LCR)

The LCR is used to control the format of the data character and is applicable to both transmitter and receiver. The LCR is read-writable. Its contents are described below.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| DLAB | Set break | Stick parity | EPS | PEN | STB | Word length select | |

| Field | Description |
|---|---|
| DLAB | This is the divisor latch access bit (DLAB) bit. When DLAB is set, access to FCR5 and the Divisor latch DLL and DLM registers is enabled. When DLAB is cleared access to the Interrupt enable register (IER), Transmitter holding register (THR) and Receiver buffer register (RBR) is enabled. |
| Set break | This is the transmission break control bit. When LCR6 is set, the transmitter serial output terminal (SOUT) is forced to the spacing state (low) and the TXEN output (RS485 mode) is held asserted (high). |
| Stick parity | This is the stick parity bit. If PEN (LCR3) is set (i.e. parity is enabled), then stick parity = 1 causes the transmission and reception of a parity bit to be in the opposite state from the value of EPS (LCR4). When EPS is set, even parity is enabled. (or cleared parity is enabled, if LCR5 is set) and when EPS is cleared, odd parity is enabled (or set parity is enabled, if LCR5 is set). This method is used to force parity to a known state. |
| EPS | This is the even parity select bit. When EPS is set, even parity is enabled. (or cleared parity is enabled, if Stick Parity - LCR5 is set). When EPS is cleared, odd parity is enabled (or set parity is enabled, if Stick Parity - LCR5 is set). |
| PEN | This is the parity enable bit. When PEN is set, a parity bit between the last data word bit and stop bit is generated in data transmitted and checked by the receiver. When PEN is cleared, no parity is selected. This bit is encoded as shown below. |

| | |
|---|---|
| STB | This bit specifies either one or two stop bits in each transmitted character. When bit 2 is cleared, one stop bit is generated in the data. When STB is set, 1½ or 2 stop bits are generated in the data: see the table below. The receiver clocks only the first stop bit, regardless of the number of stop bits selected. This bit is encoded as shown below. |
| Word length select | These two bits specify the number of bits in each transmitted or received serial character. These bits are encoded as shown below. |

**Parity select**

| LCR5:3 (binary) | Description |
|---|---|
| X X 0 | No parity |
| 0 0 1 | Odd parity |
| 0 1 1 | Even parity |
| 1 0 1 | Set parity |
| 1 1 1 | Cleared parity |

**Word length select**

| Value (binary) | Description |
|---|---|
| 0 0 | Word length is 5 bits |
| 0 1 | Word length is 6 bits |
| 1 0 | Word length is 7 bits |
| 1 1 | Word length is 8 bits |

**Stop bit length select**

| LCR2:0 (binary) | Description |
|---|---|
| 0 X X | 1 stop bit generated |
| 1 0 0 | 1½ stop bits generated |
| 1 0 1 | 2 stop bits generated |
| 1 1 0 | 2 stop bits generated |
| 1 1 1 | 2 stop bits generated |

Line break: To create a line break use the following steps. Note no invalid characters are transmitted because of the break.

1: When the THRE, transmitter holding register empty status occurs, a zero byte should be loaded.

2: After the next THRE, set the break.

3: When TEMT, transmitter empty status is set to high, wait for the transmitter to be idle.

4: Clear the break when the transmission has to be re-established.

## 9.10. Line status register (LSR)

The LSR is a read-only register that indicates the status of serial data reception.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Error in Rx FIFO | TEMT | THRE | BI | FE | PE | OE | DR |

| Field | Description |
|---|---|
| Error in Rx FIFO | In any FIFO mode, LSR7 is set when a character with a parity, framing, or break error enters the FIFO. It is cleared when the LSR is read by the host AND none of the characters in the FIFO have an error flag set. This bit is never set when FIFOs are disabled (TL16C450 mode). |
| TEMT | Transmitter empty indicator. TEMT bit is set when no character is being transmitted, AND there are no characters queued for transmission in the THR or transmit FIFO. |
| THRE | Transmitter holding register empty indicator. When FIFOs are disabled, this bit is set when the THR is empty and the UART is ready for a new character to be written to it. When FIFOs are enabled, this bit is set when the FIFO is empty. |

| | |
|---|---|
| BI | Break interrupt indicator. A “break interrupt” or “line break” is signalled when the SIN input is held low for a period of time longer than the normal transmission time for a the start, data, parity and stop bits as currently configured. The break interrupt, whatever its length, is queued like a received character whose data bits are all cleared, and a ‘BI’ flag is attached to the character in the receive FIFO. When LSR4 is set, this indicates that the next character to be read from the RBR has its BI flag set. Since the UART initially attempts to interpret the break interrupt as a received character, FE will be set because there was no valid stop bit, and the PE bit may also be set if parity bits are enabled. |
| FE | Framing error indicator. When a character is received that does not have a set bit where the stop bit is expected, a framing error flag is attached to the character in the receive FIFO. When LSR3 is set, this indicates that the next character to be read from the RBR has its FE flag set.<br>When a character is received with a framing error, the UART assumes that character synchronisation has been lost, and attempts to resynchronise by assuming that what was sampled as the stop bit of a character was actually the start bit of the next character. |
| PE | Parity error indicator. When a character is received that does not have the expected value where the parity bit is expected, a parity error flag is attached to the character in the receive FIFO. When LSR2 is set, this indicates that the next character to be read from the RBR has its PE flag set. |
| OE | Overrun error indicator. This bit is set when a character is received and there is nowhere for the received data to be stored, i.e. the RBR (TL16C450 mode) or receive FIFO (FIFO modes) was full. The received character and any associated error flags are lost. OE is cleared when the LSR is read. |
| DR | Data ready indicator. DR is set where there is any character to be read from the RBR or the receive FIFO. |

**NOTE:** Writes to the LSR are ignored, but it is recommended that they are not performed as they may have unpredictable results on other UARTs, including future versions of the BB16CF950+.

## 9.11. Interrupt enable register (IER)

The IER provides independent enable/disable controls for the four sources of interrupt from the UART. When an interrupt source is disabled, it will not cause assertion of IREQ#, and its interrupt code will not appear on the IIR.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | | Low power enable | Sleep mode enable | EMSI | ERLSI | ETHREI | ERDAI |

| Field | Description |
|---|---|
| Low power enable | This bit is read-writable, but is ignored since low power mode in the BB16CF950+ operates differently to that in the TL16C750. |
| Sleep mode enable | This bit is read-writable, but is ignored since low power mode in the BB16CF950+ operates differently to that in the TL16C750. |
| EMSI | Enable bit for ‘modem status’ interrupts. |
| ERLSI | Enable bit for ‘receiver line’ status interrupts. |
| ETHREI | Enable bit for ‘transmitter holding register empty’ interrupts. |
| ERDAI | Enable bit for ‘received data available’ interrupts, and also ‘character time-out interrupts’ in FIFO modes. |

## 9.12. Interrupt identification register (IIR)

This register indicates the interrupt status of the UART, and also some information about the FIFO status.

In order to ensure that the most time-critical interrupt sources are serviced, first, the IIR returns a code indicating the highest priority interrupt sources that is currently active. The interrupt sources are prioritised as follows:

1. (Highest priority) Receiver line status
2. Receiver character time out
3. Receiver data ready
4. Transmitter holding register empty
5. (Lowest priority) Modem status

The contents of the IIR are indicated in the table below.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| FIFOs enabled | | 750 mode | RFU | Interrupt ID | | | No interrupt |

| Field | Description |
|---|---|
| FIFOs enabled | IIR6 and IIR7 are set when FCR0 is set, i.e. the UART is in a FIFO mode. |
| 750 mode | IIR5 is set when FCR0 and FCR5 are both set, putting the BB16CF950+ into '16C750 mode'. |
| Interrupt ID | IIR3:1 identify the highest priority interrupt that is currently active, as indicated in the table below. IIR3 is always cleared when in the TL16C450 mode, because the 'character time-out' interrupt is a feature of FIFO modes only. |
| No interrupt | IIR0 is cleared when any interrupt is active, and set when there are no interrupt sources active. |

**Interrupt ID codes in IIR3:0**

| Value (binary) | Interrupt type | Priority Level | Interrupt source | To clear the interrupt source: |
|---|---|---|---|---|
| 0 1 1 0 | Receiver line status | 1 (highest) | OE, PE, FE, or BI are set in the LSR | Read the LSR |
| 1 1 0 0 | Character time-out | 2 | During the last four character times, at least one character has been waiting in the receive FIFO, and the FIFO has been inactive. | Read the RBR |
| 0 1 0 0 | Received data available | 3 | The receive FIFO has reached its trigger level. | RBR read until FIFO drops below the trigger level |
| 0 0 1 0 | THRE | 4 | THRE is set in the LSR: the UART is ready to be given more data to transmit. | Read the IIR, or write to the THR |
| 0 0 0 0 | Modem status | 5 | At least one of the MSR3:0 bits are set, because CTS#, DSR#, RI#, or DCD# have changed | Read the MSR |
| 0 0 0 1 | None | None (lowest) | No interrupt source active | - |

## 9.13. FIFO control register (FCR)

This write only register is at the same location as the interrupt identification register. It enables and clears the FIFOs, and sets the trigger level of the receiver FIFO.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Receiver Trigger | | 64 Byte FIFO Enable | RFU | DMA mode select | Transmit FIFO reset | Receiver FIFO reset | FIFO enable |

| Field | Description |
|---|---|
| Receiver Trigger | FCR6 and FCR7 set the trigger level for the receiver FIFO interrupt (see table below). |
| 64 Byte FIFO Enable | Setting this bit enables 64-byte mode of operation, and when this bit is cleared, the 16-byte mode is selected. FCR bit 5 is protected from a write, by setting the line control register (LCR) bit 7 = 1. For normal operation, LCR bit 7 is required to be cleared. This control bit is provided for backwards compatibility: if UCR1 is set then the FIFOs will be 128-deep in either case. |
| DMA mode select | This bit controls the mode of the RXRDY# and TXRDY# terminals in the TL16C750, but is ignored in the BB16CF950+. |
| Transmit FIFO reset | When setting FCR2, all the bytes in the transmitter FIFO are cleared, and the counter is reset to 0. This does not clear the multiplexer register. Writing logic 1 to this bit is self-clearing. |
| Receiver FIFO reset | When set, FCR1 clears all bytes in the receiver FIFO and resets the counter. This does not clear the multiplexer register. The logic 1 that is written to this bit position is self-clearing. |
| FIFO enable | When FCR0 is set, transmit and receive FIFOs are enabled. To enable write access to other FCR bits, this bit must be set, or they will not be programmed. An alteration to this bit clears the FIFOs. |

**Receiver FIFO Trigger Level**

| FCR 7 | FCR 6 | 16 BYTE RECEIVER FIFO TRIGGER LEVEL (BYTES) | 64 BYTE RECEIVER FIFO TRIGGER LEVEL (BYTES) | 128 BYTE RECEIVER FIFO TRIGGER LEVEL (BYTES) | RECEIVER FIFO TRIGGER LEVEL (DESCRIPTION) |
|---|---|---|---|---|---|
| 0 | 0 | 01 | 01 | 01 | Not empty |
| 0 | 1 | 04 | 16 | 32 | At least quarter-full |
| 1 | 0 | 08 | 32 | 64 | At least half-full |
| 1 | 1 | 14 | 56 | 112 | At least seven-eighths full |

## 9.14. FIFO modes

| FCR0 | FCR5 | UCR1 | IIR7:5 | UART mode | FIFO depth |
|---|---|---|---|---|---|
| 0 | X | X | 0 0 0 | TL16C450 | 1 |
| 1 | 0 | 0 | 1 1 0 | TL16C550 | 16 |
| 1 | 1 | 0 | 1 1 1 | TL16C750 | 64 |
| 1 | 0 | 1 | 1 1 0 | Deep TL16C550 | 128 |
| 1 | 1 | 1 | 1 1 1 | Deep TL16C750 | 128 |

FCR0=0: TL16C450 mode – no FIFOs for transmit or receive data, only a 1-byte holding register for each. Receive FIFO trigger level is always 1 byte.

FCR0=1 and FCR5=0 and UCR1=0: TL16C550 mode – 16-deep FIFOs for receive and transmit. Mode is identified to device driver as TL16C550 by giving IIR7:5 as 110 binary. Receive FIFO trigger levels are as stated for 16-byte FIFOs.

FCR0=1 and FCR5=1 and UCR1=0: TL16C750 mode – 64-deep FIFOs for receive and transmit. Mode is identified to device driver as TL16C750 by giving IIR7:5 as 111 binary. Receive FIFO trigger levels are as stated for 64-byte FIFOs.

FCR0=1 and UCR1=1: Deep TL16C550/750 mode – These are non-standard modes allowing the use of 128-deep FIFOs for receive and transmit. The only difference between “deep 16C550” and “deep 16C750” is the value of IIR5 and the behaviour of THRE interrupts when UCR2 is set. The modes are identified to the

device driver via the value read from IIR7:5 as TL16C550 or TL16C750 depending on whether FCR5 has been set. Receive FIFO trigger levels are as stated for 128-byte FIFOs. If UCR2 is set, the UART will attempt to make the device driver fill the 128-byte transmit FIFO by generating extra THRE interrupts (see section 9.15). Thus device drivers that support only the TL16C550 or TL16C750 UARTs may use the full 128-byte depth of the FIFOs if UCR1 and UCR2 are set via EEPROM configuration.

## 9.15. FIFO interrupt mode operation

When the receiver FIFO and receiver interrupts are enabled (FCR0 = 1, IER0 = 1, IER2 = 1), a receiver interrupt occurs as follows:

1. When the FIFO has reached its programmed trigger level, the received data available interrupt is issued to the host. This is cleared when the FIFO drops below its programmed trigger level.
2. In addition to when the FIFO trigger level is reached, and as the interrupt, is cleared when the FIFO drops below the trigger level, the IIR receive data available indication also occurs.
3. The receiver line status interrupt (IIR = 06) holds a much higher priority than the received data available (IIR = 04) interrupt.
4. When a character is transferred from the deserialiser to the receiver FIFO, the data ready bit (LSR0) is set. When the FIFO is empty, it's cleared.

When the receiver FIFO and receiver interrupts are enabled:

1. FIFO time-out interrupt occurs when the following conditions exist:
    a. A minimum of one character is still present in the FIFO.
    b. More than four continuous character times have passed (if two stop bits are programmed, the second one is included in this time delay), before a new serial character has been received.
    c. More than four continuous character times have passed, since the reading of the FIFO was carried out by the host. This causes a maximum character received to interrupt an issued delay of 160 ms at 300 baud with a 12-bit character.
2. The FIFO interrupt is cleared, as soon as a time-out interrupt has occurred. When the host reads one character from the receiver FIFO, this causes the timer to reset. The non-occurrence of a time-out interrupt, causes the time-out timer to reset after a new character is received, or after the host reads the receiver FIFO.

When the transmitter FIFO and THRE interrupt are enabled (FCR0 = 1, IER1 = 1), transmit interrupts occur as follows:

1. When the transmit FIFO is empty, it causes the transmitter holding register interrupt [IIR3:0 = 2] to occur. When either the THR is written to, or the IIR is read, this event causes the interrupt to be cleared [IIR3:0 = 1].
2. A minimum of two bytes in the transmit FIFO are required, at the same instance since the last time that THRE = 1, or this causes the transmit FIFO empty indicator (LSR5 (THRE) = 1) to be delayed one character time minus the last stop bit time. The first transmitter interrupt is instantaneous after changing FCR0 when it is enabled.

If the UART is in a "deep FIFO" mode, and UCR2 (Enable extra THRE interrupts) is set, then the transmit interrupt behaves slightly differently: the transmitter holding register interrupt [IIR3:0 = 2] still occurs when the transmit FIFO is empty. The interrupt is cleared [IIR3:0 = 1] when a) the THR is written to until less than 16 or 64 (depending on FCR5) spaces remain the transmit FIFO, or b) the IIR is read. If the THR is written and at least 16 or 64 spaces still remain in the transmit FIFO, the interrupt state returns. Note that the setting of UCR is not compatible with all device drivers, as LSR5 will still indicate a non-empty transmit FIFO when the extra THRE interrupts are generated. This may cause the device driver to fail.

The behaviour of the THRE interrupt is summarised below in terms of what events cause it to become set and cleared.

| Event | Effect on THRE interrupt |
|---|---|
| UART reset, or IER1 cleared | Interrupt cannot occur, whatever else happens, until IER1 is set |
| IER written with bit 1 set | Interrupt will be set immediately if the transmit FIFO is empty |
| Transmit FIFO becomes empty | Interrupt will be set |

| Event | Effect on THRE interrupt |
|---|---|
| IIR read | Interrupt will be cleared |
| THR written | Interrupt will be cleared |
| THR written, leaving at least 16 spaces in the transmit FIFO, AND FIFO mode is “deep 16C550”, AND UCR2 is set | Interrupt will be set |
| THR written, leaving at least 64 spaces in the transmit FIFO, AND FIFO mode is “deep 16C750”, AND UCR2 is set | Interrupt will be set |
| Number of bytes in transmit FIFO drops from TITR+1 to TITR | Interrupt will be set |

The mode with UCR2 set is intended for use with device drivers that are not BB16CF950+-aware. If a driver is being written to support the BB16CF950+, it is recommended that it should clear UCR2, and make use of the TFLR and TITR to optimise THRE interrupts.

## 9.16. FIFO polled mode operation

If any, or all of the Interrupt enable masks are cleared (IER0, IER1, IER2, IER3, or all four = 0) data and error conditions are still available from the UART by using a polling method.

The users application or driver would check transmitter, and/or receiver FIFO status by querying the Line Status Register (LSR) as described above in **Error! Reference source not found.** (**Error! Reference source not found.**)

In Polled mode, the FIFOs continue to store data in the expected fashion with the exception that “trigger level” or “time-out” flags are not generated.

## 9.17. Receive FIFO level register (RFLR)

This read-only register allows a much faster emptying of the receiver FIFO, by eliminating the need to perform an LSR read before each read of the RBR. If RFLR7 is clear, then the device driver can immediately perform N reads of the RBR, where N is the value given by RFLR6:0.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Error in Rx FIFO | Receive FIFO level | | | | | | |

| Field | Description |
|---|---|
| Error in Rx FIFO | This bit is set if any of the entries in the receiver FIFO has any of its error flags (parity, framing, or break error) set. It is cleared when the host reads the LSR and there are no subsequent errors in the FIFO. |
| Receive FIFO level | This field returns a value between 0 and 127, relating to the number of data entries stored in the receiver FIFO. The value read from this register is only valid if the UART is in a deep FIFO mode. If there are more than 127 entries in the recieve FIFO, then the number reported will be 127. |

## 9.18. Transmit FIFO level register (TFLR)

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Transmit FIFO spaces | | | | | | | |

This read-only register returns a value between 0 and 128, relating to the number of available spaces in the transmit FIFO. The value read from this register is only valid if the UART is in a deep-byte FIFO mode. The number may be greater than 128 in future devices with compatible register sets.

## 9.19. Receive FIFO flow-control trigger register (RFTR)

This register allows an arbitrary trigger level to be set for auto-RTS flow control (see section 9.25). If the value in this register is non-zero, then this value is used rather than the trigger level set by FCR7:6 (see section 9.13). Valid values are 0 to 127.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | Receive FIFO trigger level for RTS# flow control | | | | | | |

## 9.20. Receive FIFO interrupt trigger register (RITR)

This register allows an arbitrary trigger level to be set for the "received data available" interrupt (see section 9.15). If the value in this register is non-zero, then this value is used rather than the trigger level set by FCR7:6 (see section 9.13). Valid values are 0 to 127.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | Receive FIFO trigger level for "received data available" interrupt | | | | | | |

## 9.21. Transmit FIFO interrupt trigger register (TITR)

This register allows an arbitrary trigger level to be set for the "THRE" interrupt (see section 9.15). Valid values are 0 to 127. If the value in this register is non-zero, then when the number of bytes in the transmit FIFO drops from TITR+1 to TITR the THRE interrupt state will be set. The THRE interrupt will still also be generated when the transmit FIFO becomes empty, thus allowing for the case where less than TITR characters have been written to the transmit FIFO.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | Transmit FIFO trigger level for THRE interrupt | | | | | | |

## 9.22. Modem control register (MCR)

The MCR controls the interface with the modem or data set as described in Figure 20. MCR can be written and read. The RTS# and DTR# outputs are directly controlled by their control bits in this register (unless the UART is in loopback mode). A high input asserts a low signal (active) at the output terminals. The MCR bits are shown below:

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | AFE | Loop | OUT2 | OUT1 | RTS | DTR |

| Field | Description |
|---|---|
| AFE | This bit (AFE) is the auto flow-control enable. When AFE is set, the autoflow control, as described in section 9.25, is enabled. Auto flow-control may also be enabled by UCR0: see section 9.36. |
| Loop | MCR4 provides a local loopback feature for diagnostic testing of the channel. See section 9.23. |
| OUT2 | When MCR3 is set, the external serial channel interrupt is enabled. |
| OUT1 | MCR2 has no affect on operation. |
| RTS | When MCR1 is set, the RTS# output is forced low. When MCR1 is cleared, the RTS# output is forced high. The RTS# output of the serial channel can be input into an inverting line driver to obtain the proper polarity input at the modem or data set. |
| DTR | When MCR0 is set, the DTR# output is forced low. When MCR0 is cleared, the DTR# output is forced high. The DTR# output of the serial channel can be input into an inverting line driver in order to obtain the proper polarity input at the modem or data set. |

MCR7:6 are permanently cleared.

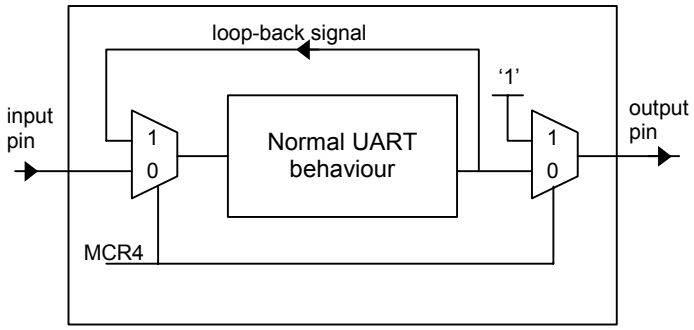
## 9.23. Loopback mode

When MCR4 is set, the UART is put into a 'loopback' mode which is useful for testing. The serial data and modem control outputs (SOUT, DTR#, RTS#, OUT1#, and OUT2#) are forced into an inactive (high) state so that attached devices are not affected by the internal testing. The five signals which would normally feed these pins are fed into the serial data and modem control inputs (SIN, CTS#, DSR#, DCD#, and RI#), which are disconnected from their input pins.

In this mode, any data transmitted is immediately received, thus allowing the host system to test the transmit and receive data paths of the UART. Interrupt control continues to operate, but based on the state of the 'looped-back' signals rather than the actual SOUT, DTR#, RTS#, OUT1#, and OUT2# input pins.

| Internal signal which is normally controlled by this input pin: | is instead controlled by the signal which normally goes to this output pin: this output is forced high. |
|---|---|
| SIN | SOUT |
| CTS# | DTR# |
| DTR# | DSR# |
| DCD# | OUT1# |
| RI# | OUT2# |




## 9.24. Modem status register (MSR)

The MSR allows the state of the modem status lines (CTS#, DSR#, RI#, and DCD#) to be read by the host. Bits 7-4 of this read-only register reflect the assertion state of the corresponding input pins, and bits 3:0 indicate whether changes have occurred on this inputs since the last time the MSR was read.

When the UART is being operated under interrupts, the host can be notified of changes in the modem status lines by enabling modem status interrupts (by setting IER3), in which case a priority-5 interrupt is generated whenever any of MSR3:0 become set. The interrupt is generated whether the setting of MSR3:0 is caused by changes on the modem status lines or by changes of MCR3:0 in loopback mode.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| DCD | RI | DSR | CTS | ΔDCD | TERI | ΔDSR | ΔCTS |

| Field | Description |
|---|---|
| DCD | Data carrier detect: set when the DCD# input is low.<br>When the UART is in loopback mode (MCR4 is set), MSR7 reflects the value last written to MCR3. |
| RI | Ring indicator: set when the RI# input is low.<br>When the UART is in loopback mode, MSR6 reflects the value last written to MCR2. |
| DSR | Data send ready: set when the DSR# input is low.<br>When the UART is in loopback mode, MSR5 reflects the value last written to MCR0. |
| CTS | Clear to send: set when the CTS# input is low.<br>When the UART is in loopback mode, MSR4 reflects the value last written to MCR1. |
| ΔDCD | ΔDCD indicates that the DCD# input has changed state since the last time the MSR was read. |
| TERI | TERI indicates that the RI# input has changed from a low to a high state since the last time the MSR was read. High-to-low transitions on RI# do not affect TERI. |
| ΔDSR | ΔDSR indicates that the DSR# input has changed state since the last time the MSR was read. |
| ΔCTS | ΔCTS indicates that the CTS# input has changed state since the last time the MSR was read. |

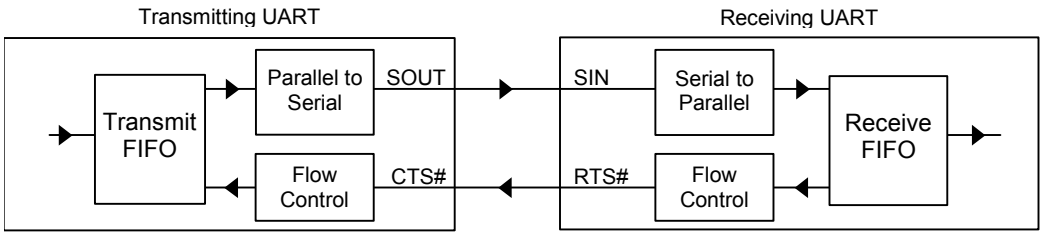
## 9.25. Auto flow control

Auto flow control consists of two functional parts: auto-CTS and auto-RTS. These features allow the flow of serial data to be throttled, preventing data loss due to receive buffer overruns, without relying on fast

interrupt service. RTS# and CTS# are often used for flow control, but if the host has to perform that function then delays in servicing interrupts can mean that the flow control is not quick enough, and data can be lost. When a pair of UARTs are connected that both have auto flow control enabled, then loss-free data transfer is possible whatever the interrupt latencies of the host systems.

The RTS# output of the receiving UART must be connected to the CTS# input of the transmitting UART, as shown below. Usually a full-duplex link will be used and so the diagram below will be only half the system, with both UARTs having a transmitting and a receiving side and being connected symmetrically.




**Auto flow control (auto-RTS and auto-CTS) example**

**auto-RTS**

Auto-RTS attempts to control the flow of serial data in to the UART. When enabled, it automatically signals to a connected device that it should stop transmitting, because there is a risk of the receive FIFO overrunning. This is done by making the state of the RTS# output dependent on the level of the receive FIFO: RTS# is asserted when the receive FIFO is below a certain trigger level, and deasserted when the receive FIFO is at or above that trigger level. The BB16CF950+ supports the setting of the trigger level to four predefined levels using FCR7:6, for compatibility with the TL16CF750, but also allows a precise value to be set using the RFTR (see section 9.19).

**auto-CTS**

Auto-CTS controls the flow of serial data out of the UART. When enabled, the transmitter will not start transmitting a new data byte unless the CTS# input is asserted. If data transmission is stopped in this way, it will restart as soon as CTS# is asserted. The UART never sends partial data bytes.

**enabling auto-RTS and auto-CTS**

It is possible to have neither auto-RTS or auto-CTS enabled (with manual control of RTS#), or both enabled, or auto-CTS enabled with RTS# held deasserted. The auto-RTS and auto-CTS modes of operation are activated by setting bits 5 and 1 of the modem control register (MCR) to 1, but see also section 9.36 for details of how auto flow control is affected by MCR1, MCR4, and UCR0.

## 9.26. Synchronisation Factor register (SFR)

The UART uses an internal clock which is faster (by a fixed integer factor) than the baud rate selected. The factor by which it is faster is called the synchronisation factor (SF). Received data is clocked into the UART every SF clock cycles, and the position of these enabled cycles is selected so as to clock in data from as close to the middle of each serial bit as possible. A higher value of SF gives a sampling time which is closer to the centre of the bits, and hence gives greater tolerance of line noise, but also gives higher power consumption and a lower maximum baud rate.

This register allows the programmer to select the synchronisation factor used. Allowed values for writing to this register are 04H, 08H, and 10H. A TL16C750 has a fixed SF of 16.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | | | SF16 | SF8 | SF4 | RFU | |

## 9.27. Programmable baud rate generator

The internal clock used by the UART's transmit and receive units is generated by taking the clock input fed into XTALI and dividing it by a 16-bit integer value (the "divisor"). The internal clock frequency is SF x the baud rate, so the divisor may be calculated as:

divisor = XTALI input frequency ÷ (desired baud rate x synchronisation factor)

If the result of this is not an integer, or is more than 65535, then the wanted baud rate is not exactly obtainable using that synchronisation factor.

The divisor is stored in two 8-bit divisor latches:

| Divisor latch most significant byte (DLM) | | | | | | | | Divisor latch least significant byte (DLL) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| Divisor value | | | | | | | | | | | | | | | |

Setting of these divisor latches is a necessary step in the initialisation of the UART before any data can be transmitted or received.

The effect of loading a divisor of zero (all bits cleared in DLL and DLM) is to stop the clocking of the receiver and transmitter, i.e. it gives a baud rate of zero. No serial data is transmitted or received, no data enters the receive FIFO or leaves the transmit FIFO. The level on SOUT can still be changed by writing LCR6, and all other UART functions continue to operate.

## 9.28. Half-duplex control register (HDCR)

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Hold CTS | Half-duplex mode | | |

Bits 0-2 control the driving of the enable lines TXEN and RXEN for the line drivers and receivers as follows:

**Half-duplex mode – description of settings**

| Value (binary) | Description |
|---|---|
| 0 0 0 | RS485 half-duplex, autogating mode |
| 0 0 1 | RS485 half-duplex, RTS controlled: transmitter enabled, receiver disabled when RTS asserted, and transmitter disabled, receiver enabled when RTS deasserted. |
| 0 1 0 | RS485 half-duplex, DTR controlled: transmitter enabled, receiver disabled when DTR asserted, and transmitter disabled, receiver enabled when DTR deasserted. |
| 0 1 1 | RS485 half-duplex, RTS controlled: transmitter disabled, receiver enabled when RTS asserted, and transmitter enabled, receiver disabled when RTS deasserted. |
| 1 0 0 | transmitter and receiver both disabled |
| 1 0 1 | transmitter enabled, receiver disabled (RS485 half duplex, manual Tx) |
| 1 1 0 | transmitter disabled, receiver enabled (RS485 half duplex, manual Rx) |
| 1 1 1 | transmitter and receiver both enabled (RS422/485 full duplex) |

**Half-duplex mode – logical definition of settings**

| Value (binary) | Transmitting data or line break | RTS asserted | DTR asserted | TXEN | RXEN |
|---|---|---|---|---|---|
| 0 0 0 | Y<br>N | | | 1<br>0 | 0<br>1 |
| 0 0 1 | | Y<br>N | | 1<br>0 | 0<br>1 |
| 0 1 0 | | | Y<br>N | 1<br>0 | 0<br>1 |
| 0 1 1 | | Y<br>N | | 0<br>1 | 1<br>0 |
| 1 0 0 | | | | 0 | 0 |
| 1 0 1 | | | | 1 | 0 |
| 1 1 0 | | | | 0 | 1 |
| 1 1 1 | | | | 1 | 1 |

Bit 3 of this register allows the CTS input of the UART to be held in a ‘true’ state, regardless of the state of the CTS# input pin:

**Hold CTS**

| Value | Description |
|---|---|
| 0 | CTS reflects state of CTS# pin |
| 1 | CTS held true |

## 9.29. Autogating

Autogating is a feature that automatically controls TXEN and RXEN for half-duplex operation, putting the transceivers in receive mode for as much of the time as possible.

- If a data byte, or its start, parity or stop bits are being transmitted, then TXEN is driven high and RXEN is driven low. TXEN is asserted, and RXEN deasserted, two XTALI clock cycles before SIN goes low for the start of the transmission. This is to allow for the enable time of the RS485 driver, which can be significant at high baud rates.
- If LCR6 (Set Break) is set, implying the output of a line break state, then TXEN is driven high and RXEN is driven low.
- If neither of the above apply, then TXEN is driven low and RXEN is driven high.

## 9.30. Chip type identification

To determine whether a UART is a BB16CF950+, and to clear the extended registers safety catch:

- Ensure that bit 4 of IER is cleared
- Write 80h to the LCR
- Set X=23h
- Repeat the following 42 times:
    - Write the value X to the DLM
    - Set X=X x 2
    - Set bit 0 of X to the exclusive-or of bit 7 and bit 6 of X
- If IER4 is now set, the chip is identified as a BB16CF950+, or a future device with the same extended-register system. Write a value of 2 to the IRSR, and then read the CIDR to identify the exact device type. If IER4 is clear, the chip is not a Brain Boxes device.

Do not make any other read or write access to the UART while writing the sequence of values to the DLM. For reference, the sequence should be: 23h, 47h, 8Fh, 1Eh, 3Ch, 79h, F2h, E4h, C8h, 91h, 22h, 45h, 8Bh, 16h, 2Ch, 59h, B3h, 67h, CEh, 9Dh, 3Ah, 75h, EAh, D4h, A9h, 53h, A7h, 4Fh, 9Fh, 3Eh, 7Dh, FAh, F4h, E8h, D0h, A1h, 43h, 87h, 0Eh, 1Ch, 38h, 71h.

## 9.31. Extended registers safety catch

Some host systems may violate the TL16Cx50 specifications and write to the MSR, which will set the IRSR in the BB16CF950+. If they then attempt to access the scratch register, they may cause failure due to an apparently failed scratch register or due to corruption of the extended registers. If this is found to be a problem, then the IRSR may be protected by setting the ERSC bit in the configuration EEPROM (see section 6.1). If this EEPROM bit is set, then after any reset of the UART a safety-catch will be engaged which causes writes to the MSR/IRSR address to be ignored. To clear the safety catch and enable access to the extended register set, a BB16CF950+-aware device driver may perform the chip type identification sequence (section 9.30) which will enable writes to the IRSR until the next UART reset.

## 9.32. Chip ID register (CIDR)

This read only register provides an identification code to enable software to distinguish between versions of the BB16CF950+, or future components with a similar interface. The only code currently defined is 01H, which indicates the first revision of the BB16CF950+.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Chip identification code | | | | | | | |

**Chip identification code**

| Value | Description |
|---|---|
| 01h | BB16CF950+, first revision |
| 02-0Fh | Reserved for devices with register sets compatible with the BB16CF950+. Device drivers may use assume the device conforms with this specification. |
| others | RFU |

## 9.33. USER0 control register (USR0R)

This read-write register, whose reset value is taken from the configuration EEPROM, controls the operation of the USER0 input/output pin.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | | | | Read state | Invert on reset | Set state | Output enable |

| Field | Description |
|---|---|
| Read state | When read, this bit shows the current state of the USER0 pin. Writes to this bit are ignored. |
| Invert on reset | If this bit is set and USR0R0 is set, the USER0 output is inverted (compared to the value of USR0R1) whenever the host resets the device, either using the RESET line or by setting COR7. |
| Set/read state | The value written to this bit sets the value driven onto the USER0 pin when USR0R0 is set. |
| Output enable | When this bit is set, the USER0 pin is driven by the BB16CF950+. When cleared, the USER0 pin is placed in a high-impedance state and may be used as an input. |

**Examples of settings for USR0R2:0**

| Value (binary) | Description |
|---|---|
| X X 0 | User input: read USR0R3 to get current value |
| 0 0 1 | User output, driven low |
| 0 1 1 | User output, driven high |
| 1 0 1 | Reset output, active high |
| 1 1 1 | Reset output, active low |

NOTE:

1: Between the power-up of the device and the assertion of the host RESET signal, the value of USR0R will be indeterminate: it may be driving high, driving low or tristated. Some protection may be required if USER0 is to be used as an input.

2: Between the assertion of the host RESET signal and the reading of the EEPROM, USR0R will be set to 00h.

## 9.34. USER1 control register (USR1R)

This read-write register, whose reset value is taken from the configuration EEPROM, controls the operation of the USER1 output, in modes where it is available. In modes where USER1 is not available, the value of this register is ignored.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | | | | Read state | Output clock | Set state | Output enable |

| Field | Description |
|---|---|
| Read state | When read, this bit shows the current state of the USER0 pin. Writes to this bit are ignored. |
| Enable clock output | If this bit is set and USR1R0 is set, the USER1 output is driven with a copy of the signal on the XTALI pin. |
| Set/read state | The value written to this bit sets the value driven onto the USER1 pin when USR1R0 is set. |
| Output enable | When this bit is set, the USER0 pin is driven by the BB16CF950+. When cleared, the USER0 pin is placed in a high-impedance state and may be used as an input. |

**Examples of settings for USR1R2:0**

| Value (binary) | Description |
|---|---|
| X X 0 | User input: read USR1R3 to get current value |
| 0 0 1 | User output, driven low |
| 0 1 1 | User output, driven high |
| 1 X 1 | Clock output |

NOTE:

1: Between the power-up of the device and the assertion of the host RESET signal, the value of USR1R will be indeterminate: it may be driving high, driving low or tristated. Some protection may be required if USER1 is to be used as an input.

2: Between the assertion of the host RESET signal and the reading of the EEPROM, USR1R will be set to 00h.

## 9.35. Bluetooth power control register (BPR)

This read-write register, whose reset value is taken from the configuration EEPROM, controls the PWRON output pin in Bluetooth mode. PWRON will be in a high state unless any of the conditions enabled by BPR3:0 occur. PWRON is always high during the EEPROM-reading phase following a hardware reset. This is so that the SBDIO pin, which drives both the EEPROM and the BlueCore, may safely be driven.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | | Doing auto-reset (RO) | Enable auto-reset | CSR4 power down | CSR2 power down | COR power down | Power down now |

| Field | Description |
|---|---|
| Doing auto-reset | This read-only bit indicates whether an auto-reset count is in progress. This bit will be set for $2^{18}$ XTALIN clock cycles after any of the events enabled by BPR3:0 have changed PWRON from low to high, if BPR4 is set. |
| Enable auto-reset | If this bit is set and USER0 is configured to be a reset output, then USER0 will be asserted for $2^{18}$ XTALIN clock cycles after any high→low transition on PWRON. No BB16CF950+ registers or flags are reset, but during the reset time any serial transmission or reception taking place will be corrupted. |
| CSR4 power down | If this bit is set, PWRON will be driven low whenever CSR bit 4, XE#, is set by the host. |
| CSR2 power down | If this bit is set, PWRON will be driven low whenever CSR bit 2, PwrDwn, is set by the host. |

| | |
|---|---|
| COR power down | If this bit is set, PWRON will be driven low whenever the function is not enabled for I/O access. |
| Power down now | If this bit is set, PWRON will be driven low immediately. This bit is provided for device-driver control of the PWRON pin. |

If enabled by BPR4, auto-reset will be triggered by a hardware reset of the BB16CF950+, so that the external device is still reset if its power-up is due to a system power-up rather than a change of state of the PWRON pin.

## 9.36. UART configuration register (UCR)

This read-write register, whose reset value is taken from the configuration EEPROM, configures features of the UART that are not present in the TL16C750.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| RFU | | | | Enable low-power mode | Enable extra THRE interrupts | Enable deep FIFOs | Force AFC on |

| Field | Description |
|---|---|
| Enable low-power mode | When this bit is set, UART clock cycles are skipped wherever possible to save power. When this bit is reset, the UART clock runs continuously. |
| Enable extra THRE interrupts | When this bit is set and the UART is in “Deep 16C550” mode, the UART will keep generating transmit-buffer-empty interrupts until there are less than 16 spaces left in the transmit FIFO, or no data is written. |
| Enable deep FIFOs | When this bit is set, the FIFOs are always 128 deep when FCR0 is set. |
| Force AFC on | When this bit is set, and MCR4 is cleared, auto flow control is always on, even when the driver doesn’t enable it (this should only be used when it is known that the attached device will be using RTS-CTS handshaking for flow control) |

Operation when UCR0 is cleared (standard TL16C750 operation):

| MCR5 (AFE) | MCR1 (RTS) | Flow control configuration |
|---|---|---|
| 1 | 1 | Auto-RTS# and auto-CTS# enabled (auto flow control enabled) |
| 1 | 0 | Auto-CTS# only enabled |
| 0 | X | Auto-RTS# and auto-CTS# disabled |

Operation when UCR0 is set:

| MCR4 (Loop) | MCR5 (AFE) | MCR1 (RTS) | Flow control configuration |
|---|---|---|---|
| 0 | X | 1 | Auto-RTS# and auto-CTS# enabled (auto flow control enabled) |
| 0 | X | 0 | Auto-CTS# only enabled, RTS# deasserted |
| 1 | 1 | 1 | Auto-RTS# and auto-CTS# enabled (auto flow control enabled) |
| 1 | 1 | 0 | Auto-CTS# only enabled |
| 1 | 0 | X | Auto-RTS# and auto-CTS# disabled |

## 9.37. Mode and features register (MFR)

This register configures the mode of operation of the BB16CF950+, and features of its host interface. It is initialised from the serial EEPROM and will probably not need to be changed by host I/O accesses.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| Enable ESR | Wake on interrupt | 0 | 0 | 0 | 0 | Mode | |

| Field | Description |
|---|---|
| Enable ESR | This bit enables the ESR register to be accessible in attribute memory space, allowing status changes to be reported. This bit should not be set in CompactFlash cards, where this register is not defined. |
| Wake on interrupt | When this bit is set, an interrupt event will cause a status change, with similar behaviour to the Ring Indicator status change. |
| Mode | Selects the usage mode of the BB16CF950+, as listed in the table below. |

**Mode field**

| Value (binary) | Description |
|---|---|
| 0 0 | The BB16CF950+ is in Bluetooth mode. |
| 0 1 | The BB16CF950+ is in RS232 mode. |
| 1 0 | The BB16CF950+ is in RS485 mode. |
| 1 1 | The BB16CF950+ is in local bus mode. |

## 9.38. Scratchpad register (SCR)

The scratch register is an 8-bit read/write register that has no affect on either channel in the UART. It is intended to be used by the programmer to hold data temporarily.

## 9.39. Inverted scratchpad register (ISCR)

Writes to this register set the scratchpad register. Reads of this register return the current scratch register value with all bits inverted.

## 9.40. Serial Bus control register (SBR)

The BB16CF950+ controls the EECS, EECK and SBDIO pins after a deassertion of the host bus RESET signal, in order to read in configuration and CIS data. Once this is complete, these signals may be controlled though I/O accesses to this register. The SPI interface (Bluetooth mode) is controlled using the same register.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|
| SPI_CSB | SPI_CLK | RFU | MISO (RO) | EECS | EECK | SBDIO output enable | SBDIO data in/out |

| Field | Description |
|---|---|
| SPI_CSB | In Bluetooth mode, this bit controls the SPI_CSB# pin – the pin is asserted (low) when the bit is set. In other modes its value is ignored. |
| SPI_CLK | In Bluetooth mode, this bit controls the SPI_CLK pin. In other modes its value is ignored. |
| MISO | In Bluetooth mode, this bit reports the level on the MISO pin. In other modes it is always cleared. |
| EECS | This bit directly controls the EECS output. |
| EECK | This bit directly controls the EECK output. |
| SBDIO output enable | When this bit is set, the value last written to SBR0 is driven on the SBDIO pin. When it is cleared, the SBDIO pin is tristated. |
| SBDIO data in/out | When read, this bit returns the current logic level on the SBDIO pin.<br>When written, this bit controls the logic level driven onto SBDIO when SBR1 is set. |

# 10. Factory programming mode

The BB16CF950+ is designed to allow access to the control signals of the serial EEPROM and the SPI bus through the host bus connector. This to allow factory programming of components on the same PCB, using a special programming jig.

The factory programming mode is entered when a certain combination of host bus signals is recognized. This combination is not seen in usual operation.

**Signal levels needed to enter factory programming mode**

| Host bus signal | Value |
|---|---|
| WE# | 0 |
| OE# | 0 |
| REG# | 1 |
| RESET | 0 |

When in factory programming mode, the BB16CF950+ drives the control signals of the serial EEPROM and the SPI bus with values of certain signals from the host bus, and drives two of the host bus outputs with the EEPROM and SPI outputs.

**Host bus input signals used in factory programming mode**

| Host bus signal | Drives | Bluetooth application |
|---|---|---|
| A1 | USER0 | Reset |
| A2 | LI7 | SPI CSB# |
| A3 | SBDIO output value | SPI MOSI |
| A4 | LI5 | SPI CLK |
| A5 | LI6 (inverted) | PWRON (inverted) |
| A6 | SBDIO output disable | hold low for SPI access |
| A7 | EECK | |
| CE1# | EECS | |

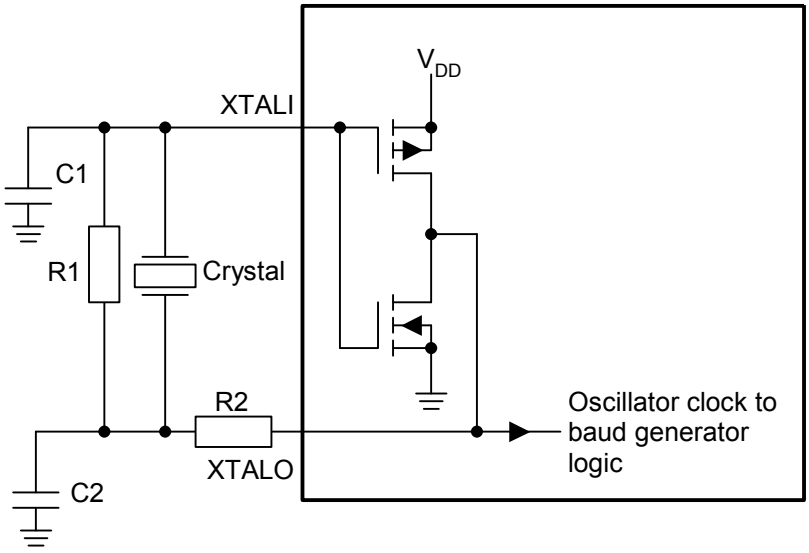
**Host bus output signals used in factory programming mode**

| Host bus signal | Driven with level on | Bluetooth application |
|---|---|---|
| IREQ# | LI4 | MISO |
| INPACK# | SBDIO | |

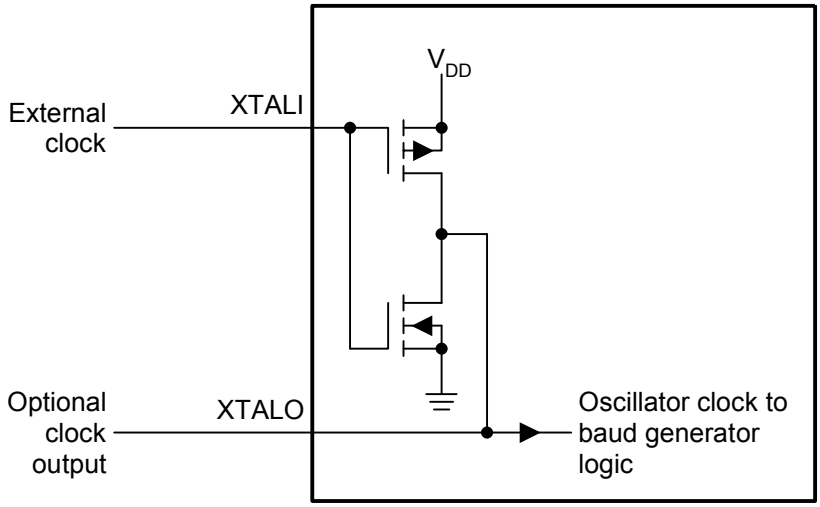
# 11. Clock/oscillator pins

The BB16CF950+ provides a clock input and a logically inverted output suitable for driving a crystal oscillator as shown below:

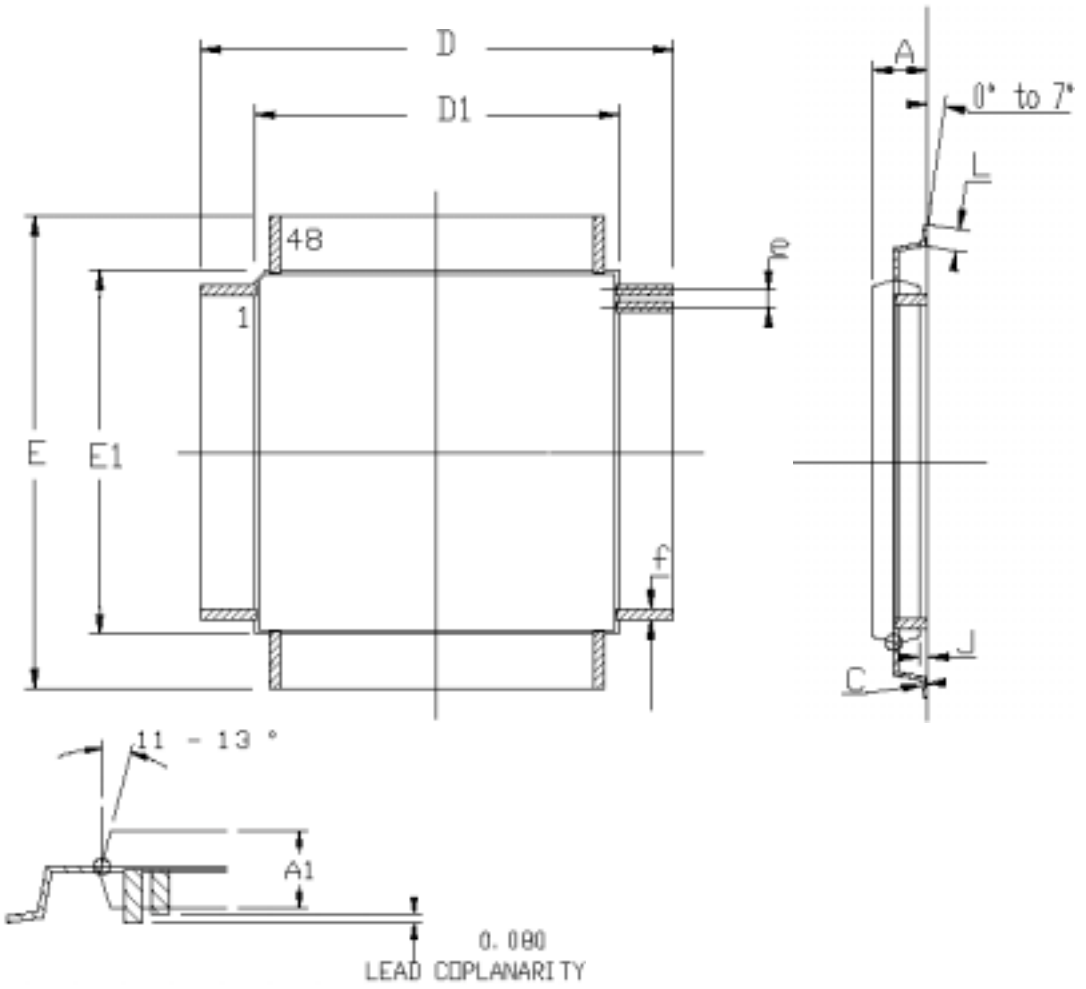



Alternatively, the XTALI pin may be driven from an external clock signal, and the XTALO pin used as an optional clock output.




# 12. Package

The package is a standard VQ48 package, with 0.5 mm lead pitch, and a 1.2 mm max height from PCB.




| | Min | Max | BSC | unit |
|---|---|---|---|---|
| A | - | 1.20 | - | mm |
| A1 | 0.95 | 1.05 | - | mm |
| C | 0.09 | 0.20 | - | mm |
| D | - | - | 9.00 | mm |
| D1 | - | - | 7.00 | mm |
| E | - | - | 9.00 | mm |
| E1 | - | - | 7.00 | mm |
| J | 0.05 | 0.15 | - | mm |
| L | 0.45 | 0.75 | - | mm |
| e | - | - | 0.50 | mm |
| f | 0.17 | 0.27 | - | mm |

# 13. Glossary

BPR Bluetooth power control register
BSC Basic spacing between centres
CIDR Chip ID register
COR Configuration option register
CSR Card status register
DLL LSB divisor latch
DLM MSB divisor latch
ERSR Extended register safety catch
ESR Extended status register
FCR In CF interface: Function Control Registers
FCR In UART: FIFO Control Register
HDCR Half-duplex control register
IER Interrupt enable register
IIR Interrupt identification register
IRSR Indexed register select register
ISCR Inverted scratch pad register
LCR Line control register
LSR Line status register
MCR Modem control register
MFR Mode and features register
MSR Modem status register
PRR Pin replacement register
RBR Receiver buffer register
RFLR Receive FIFO level register
RFTR Receive FIFO flow-control trigger register
RFU Reserved for future use: register bits described as RFU should be cleared during writes, and ignored during reads. They will be clear when read, but for future compatibility this should not be assumed.
RITR Receive FIFO interrupt trigger register
RO Read-only
SBR Microwire and SPI control register
SCR Scratch pad register
SF Synchronization Factor
SFR Synchronisation factor register
TFLR Transmit FIFO level register
THR Transmitter holding register
TITR Transmit FIFO interrupt trigger register
UCR UART configuration register
USR0R USER0 control register
USR1R USER1 control register
VQ48 48-pin VQFP package
VQFP Very thin Quad Flat Pack

# Version History

| Version | Date | Author | Checked By | Comments |
|---|---|---|---|---|
| 0.91 | 26/9/2001 | Michael Attenborough | | Imported text into standard confidential document template. |
| 0.92 | 22/10/2001 | Michael Attenborough | | Various minor corrections, clarifications and completions. Now nominally complete, except for UART operation timings. |
| 0.94 | 27/11/2001 | Michael Attenborough | | Added AFE bit to MCR description.<br>Corrections to register summary table.<br>Noted that CSR2 and 4 are not ignored<br>MCR5 documented<br>Behaviour when divisor=0 documented<br>Other minor corrections |
| 0.96 | | Michael Attenborough | | Added specification of extended register safety catch.<br>Added clarifications and corrections to description of Line Status Register.<br>Added RFTR, RITR and TITR registers. |
| 0.98 | | Michael Attenborough | | Added diagram and description for use with BlueCore02.<br>Changed USR0R and USR1R to have separate set and read value bits.<br>Removed BTMODE pin, added LIVIO pin.<br>Autogating starts two clock cycles before transmission.<br>UCR1 now enables 128-byte FIFOs. |
| 1.00 | 25/4/2002 | Michael Attenborough | | Device named BB16CF950+. |
| 1.02 | 10/9/2002 | Michael Attenborough | | Used text from CompactFlash specification 1.4 to describe host bus signals, COR, CSR and PRR.<br>Inserted static timing results and I/O parameters.<br>Reworded UART description. |
| 1.04 | 8/10/2002 | Michael Attenborough | | Added note on LIVIO ramp-up to section 5.1. |