External clock clock (pin C9)

The external clock functions as the source clock for the entire processor. On start-up, the processor runs at the same rate as the external clock. The processor program may access the housekeeping SPI to set the processor into PLL mode or DCO free-running mode. In PLL mode, the external clock is multiplied up by the feedback divider value to obtain the core clock. In DCO mode, the processor is driven by a trimmed free-running ring oscillator.

UART ser tx (pin F7) and ser rx (pin E7)

The UART is a standard 2-pin serial interface that can communicate with most similar interfaces at a fixed baud rate. Although the UART operates independently of the CPU, data transfers are blocking operations which will generate CPU wait states until the data transfer is completed.

The behavior of the UART can be modified by changing values in the registers below:

Table 11 **`reg_uart_clkdiv`**

0x20000003								**0x20000002**								**0x20000001**								**0x20000000**								address
UART clock divider																																value
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	bit

The entire 32 bit word encodes the number of CPU core cycles to divide down to get the UART data bit rate (baud rate). The default value is 1.

Example: If the external crystal is 12.5MHz, then the core CPU clock runs at 100MHz.
To get 9600 baud, 100E6 / 9600 = 10417 (hex value 0x28b1).

Table 12 **`reg_uart_data`**

0x20000007								0x20000006								0x20000005								**0x20000004**								address
(unused, value is 0x0)																																value
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	bit

Writing a value to this register will immediately start a data transfer on the SER_TX pin. If a UART write operation is pending, then the CPU will be blocked with wait states until the transfer is complete before starting the new write operation. This makes the UART transmit a relatively expensive operation on the CPU, but avoids the necessity of buffering data and checking for buffer overflow. Reading a value from this register returns 255 (0xff) if no valid data byte is in the receive buffer, and returns the value of the receive buffer otherwise, and clears the receive buffer for additional reads. Note that there is no FIFO associated with the UART.

Table 13 **`reg_uart_enable`**

0x2000000b								0x2000000a								0x20000009								**0x20000008**								address
(unused, value is 0x0)																																value
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	bit

The UART must be enabled to run (default disabled)