

1. INTRODUCTION

With the proliferation of portable electronic devices, power efficient data transmission has become increasingly important. For serial data transfer, universal asynchronous receiver / transmitter (UART) circuits are often implemented because of their inherent design simplicity and application specific versatility. Components such as laptop keyboards, palm pilot organizers and modems are few examples of devices that employ UART circuits. In this work, design and analysis of a robust UART architecture has been carried out to minimize power consumption during both idle and continuous modes of operation.

UART, an introduction

An UART (universal asynchronous receiver / transmitter) is responsible for performing the main task in serial communications with computers. The device changes incoming parallel information to serial data which can be sent on a communication line. A second UART can be used to receive the information. The UART performs all the tasks, timing, parity checking, etc. needed for the communication. The only extra devices attached are line driver chips capable of transforming the TTL level signals to line voltages and vice versa.

To use the device in different environments, registers are accessible to set or review the communication parameters. Setable parameters are for example the communication speed, the type of parity check, and the way incoming information is signaled to the running software.

UART types

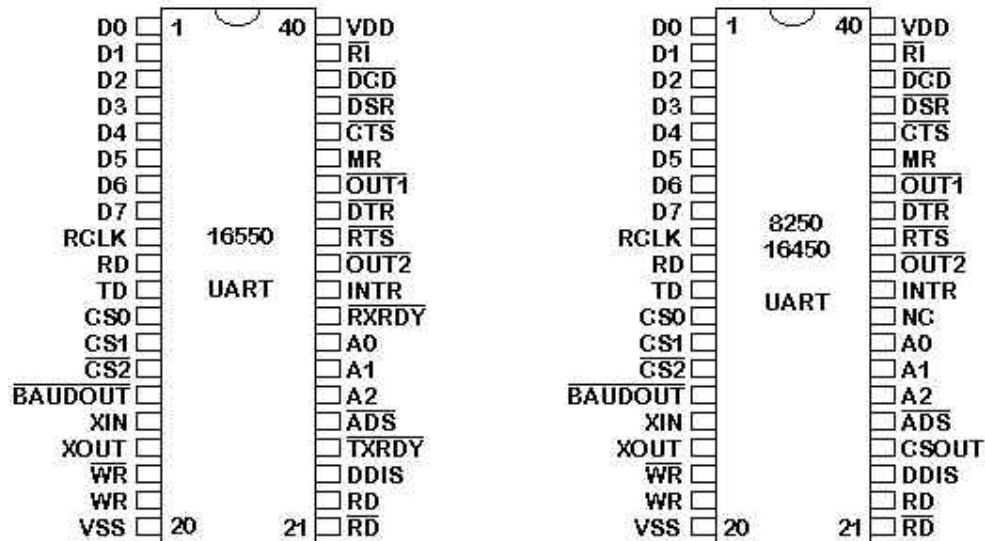
Serial communication on PC compatibles started with the 8250 UART in the XT. In the years after, new family members were introduced like the 8250A and 8250B revisions and the 16450. The last one was first implemented in the AT. The higher bus speed in this computer could not be reached by the 8250 series. The differences between these first UART series were rather minor. The most important property changed with each new release was the maximum allowed speed at the processor bus side.

The 16450 was capable of handling a communication speed of 38.4 kbs without problems. The demand for higher speeds led to the development of newer series which would be able to release the main processor from some of its tasks. The main problem with the original series was the need to perform a software action for each single byte to transmit or receive. To overcome this problem, the 16550 was released which contained two on-board FIFO buffers, each capable of storing 16 bytes. One buffer for incoming, and one buffer for outgoing bytes.

A marvelous idea, but it didn't work out that way. The 16550 chip contained a firmware bug which made it impossible to use the buffers. The 16550A which appeared soon after was the first UART which was able to use its FIFO buffers. This made it possible to increase maximum reliable communication speeds to 115.2 kbs. This speed was necessary to use effectively modems with on-board compression. A further enhancement introduced with the 16550 was the ability to use DMA, direct memory access for the data transfer. Two pins were redefined for this purpose. DMA transfer

is not used with most applications. Only special serial I/O boards with a high number of ports contain sometimes the necessary extra circuitry to make this feature work. The 16550A is the most common UART at this moment. Newer versions are under development, including the 16650 which contains two 32 byte FIFO's and on board support for software flow control. Texas Instruments is developing the 16750 which contains 64 byte FIFO's.

The pin diagram of the 16550 UART along with the previous 8250 and 16650 are given below.



The 16550 is chip compatible with the 8250 & 16450. The only two differences are pins 24 & 29. On the 8250 Pin 24 was chip select out which functioned only as an indicator to if the chip was active or not. Pin 29 was not connected on the 8250/16450 UARTs. The 16550 introduced two new pins in their place. These are Transmit Ready and Receive Ready which can be implemented with DMA (Direct Memory Access). These Pins have two different modes of operation. Mode 0 supports single transfer DMA where as Mode 1 supports Multi-transfer DMA.

Mode 0 is also called the 16450 mode. This mode is selected when the FIFO buffers are disabled via Bit 0 of the FIFO Control Register or When the FIFO buffers are enabled but DMA Mode Select = 0. (Bit 3 of FCR) In this mode RXRDY is active low when at least one character (Byte) is present in the Receiver Buffer. RXRDY will go inactive high when no more characters are left in the Receiver Buffer. TXRDY will be active low when there are no characters in the Transmit Buffer. It will go inactive high after the first character / byte is loaded into the Transmit Buffer.

Mode 1 is when the FIFO buffers are active and the DMA Mode Select = 1. In Mode 1, RXRDY will go active low when the trigger level is reached or when 16550 Time Out occurs and will return to inactive state when no more characters are left in the FIFO. TXRDY will be active when no characters are present in the Transmit Buffer and will go inactive when the FIFO Transmit Buffer is completely Full.

All the UARTs pins are TTL compatible. That includes TD, RD, RI, DCD, DSR, CTS, DTR and RTS which all interface into your serial plug, typically a D-type connector. Therefore RS232 Level Converters (which we talk about in detail later) are used. These are commonly the DS1489 Receiver and the DS1488 as the PC has +12 and -12 volt rails which can be used by these devices. The RS232 Converters will convert the TTL signal into RS232 Logic Levels.

2. UART DESIGN AND PROTOCOL

Our basic UART design consists of four main components: two counters, a shifter and a finite state machine which is shown in figure 1. The system is initialized in the start state with P_0 through P_9 (10 bit group) of the shifter preset to values of '1'. The first two components CNT12 and CNT16, divide the clock frequency down to the desired data transmission baud rate.

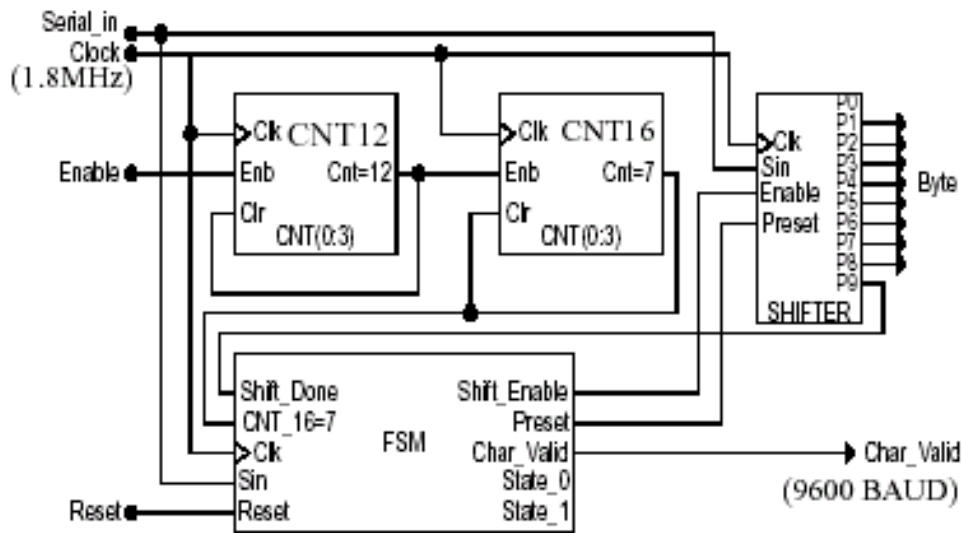


Figure 1. Serial interface UART schematic

The finite state machine (FSM) is initially in the start state where it waits for a falling edge of the serial in data to signal in the start of an incoming 10-bit word. The FSM then transitions to a second state where it waits for a

pulse from the CNT16 counter. CNT16 sends its first carry out pulse on its eighth count and remaining pulse signals every sixteen counts. This makes the transmission rate equal to $CLK/196$ ($12 \times 16 = 196$). When the signal is received, the FSM enables the shift register to sample the serial in signal. In this manner, the data is sampled (and shifted in) at the middle of each bit window (figure 2). Since by protocol convention, the first bit is always '0', 10-bit-shifted-in completion signal (char value) can be generated when P_9 changes from '1' to '0' (figure 2).

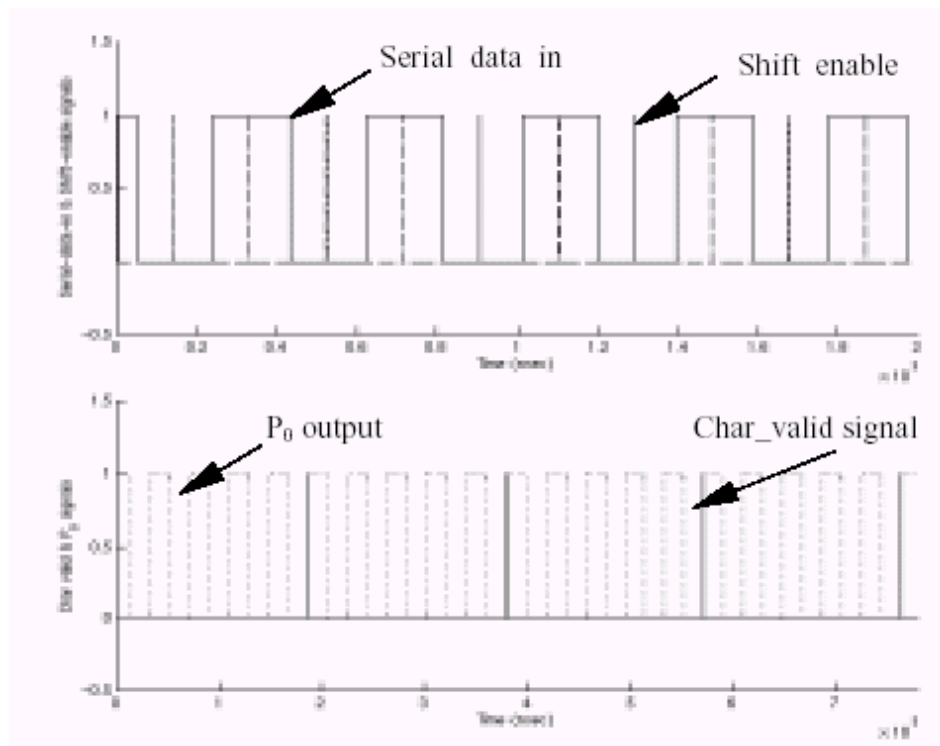


Figure 2. UART Output waveforms.

3. RESULTS AND DISCUSSIONS

3.1 Voltage – Scaling

Design optimization was carried out with the primary goal of reducing the UART power consumption during both continuous and idle modes of operation. The first approach used to minimize power consumption was to scale the circuit operating voltage (V_{dd}). The power consumption of CMOS circuits can be approximated by equation 1 where α is the activity factor, f_{clk} is the operating frequency, C_L is the load capacitance and V_{dd} is the supply voltage.

$$P \approx \alpha f_{clk} C_L V_{dd}^2$$

Here it is assumed that the voltage swing of logic is the same as the supply voltage, which is infact the most common case. According to equation 1, V_{dd} scaling is the most efficient way to reduce power.

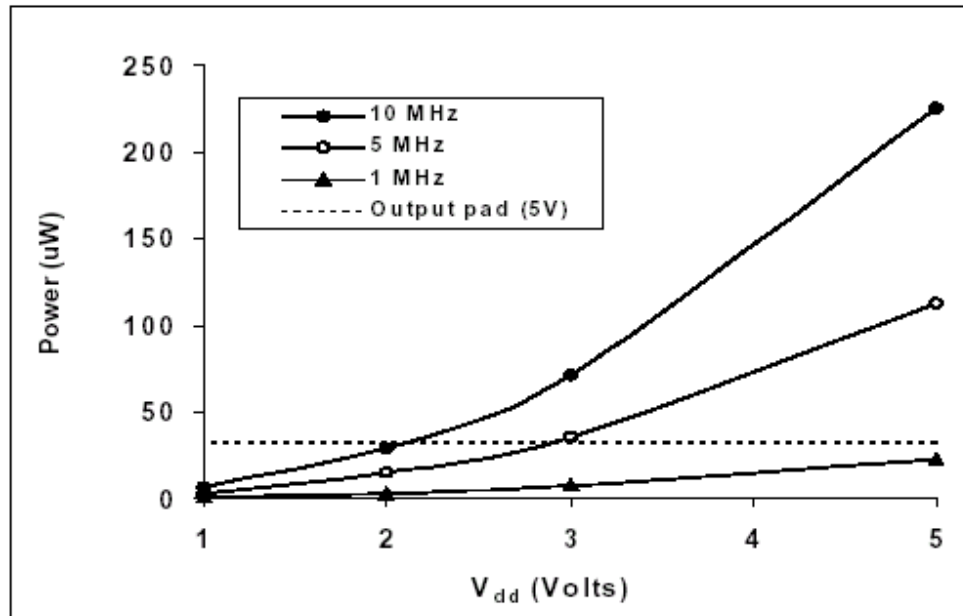


Figure 3. Logic Power vs. Supply Voltage for different operation frequency.

Figure 3 shows where the logic power scales quadratically with supply voltage and linearly with operating frequency. The logic power consumption is comparable for both continuous and idle modes of operation. Output pad power consumption was modeled as the P_o signal driving a large load capacitance (50pF) since the UART logic is almost identical for transmitting or receiving data. For the simulation the output pad voltage was fixed at 5volts power supply.

3.2 Clock – gating

To reduce the logic power in idle mode, a clock gating technique was employed. Since a large amount of power was consumed by the counter elements in the idle mode, the UART design was modified so that both the

counters received a gated clock signal controlled by FSM outputs. The original clock signal was fed into a two input AND gate and the other input was the control signal from the FSM. Figure 4 shows that, with clock gating there is approximately a 78% reduction in the idle mode power consumption. However, there is also a 33% increase in circuit power consumption during continuous mode operation. This corresponds to the logic overhead associated with the clock gating circuitry that has a unity activity factor.

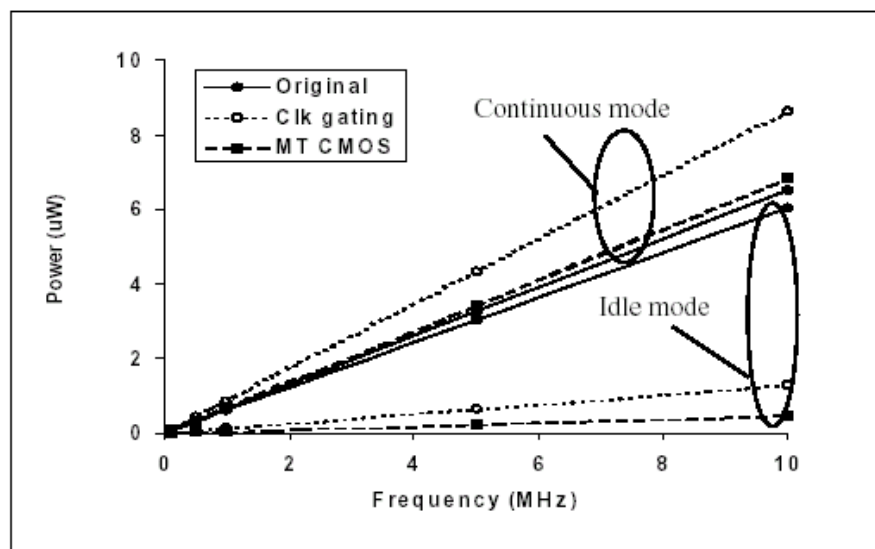


Figure 4 Logic Power vs. Operating Frequency in various low power schemes.

3.3 Multi-threshold CMOS (MT – CMOS)

As supply voltage was scaled down to 1 volt, threshold voltages of most devices were also scaled down to maintain performance. Since leakage current is larger in low V_t devices, a MT – CMOS approach was investigated.

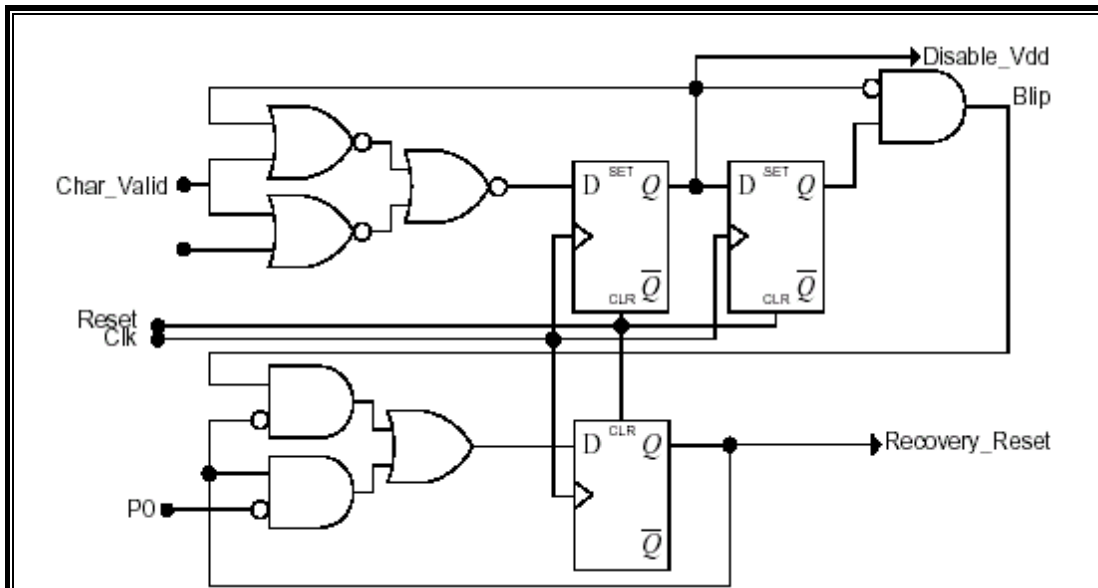


Figure 5. Disable_Vdd signal generator for MT-CMOS and Vdd-cutoff.

To implement a MT – CMOS design, a high V_t PMOS was inserted between V_{dd} (1volt) and the power supply rail. Figure 5 shows the circuitry that generates two important signals – a disable V_{dd} signal which controls the gate of high V_t PMOS, and a self timed reset signal which ensures that the UART is reset after recovering from sleep mode.

The circuit in figure 5 detects a falling edge on the serial line, which signifies the start of an incoming byte (figure 6). The high V_t PMOS transistor is turned ON by making disable V_{dd} signal low. Then, a synchronizer circuitry that generates a blip signal could be used to reset the FSM and the shifter. However, at high frequency (greater than 5MHz) the blip pulse is not wide enough for all nodes to charge. This could be solved via a cascade of shift registers to extend the width, but the objective is to use minimal overhead for all operational frequencies. Therefore a self-timed reset circuitry is designed that latches the blip signal and generates a variable width pulse signal,

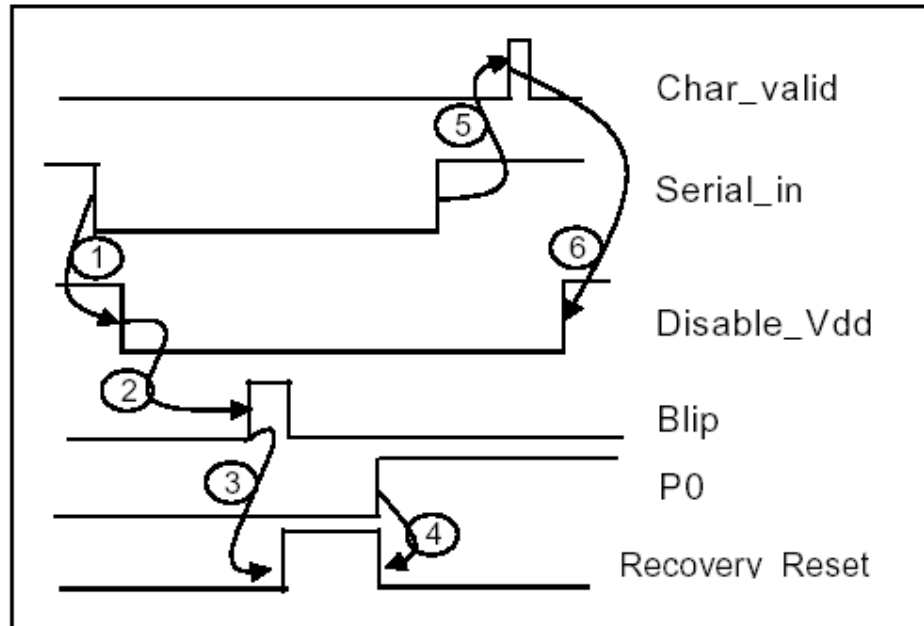


Figure 6. Timing diagram for Disable_Vdd signal generator.

Recovery Reset. The pulse width of Recovery Reset directly depends on the P_0 signal. After every transmission, the P_0 signal is completely discharged since it represents the stop bit (0 byte protocol convention). Therefore, while resetting the UART, the pulse width of Recovery Reset is extended until P_0 signal is 1. This self-timed approach ensures proper operation of the UART with MT-CMOS design at higher frequencies.

Using the disabled V_{dd} signal generator with MT – CMOS technique a circuit recovery time of $1\mu\text{second}$ (average) was measured (figure 7). The maximum allowable recovery time (when V_{dd} tmp mode is fully charged to the supply voltage after the disabled V_{dd} signal goes low) is $9.6\mu\text{second}$ for 10MHz operation frequency. It was seen that the MT – CMOS approach improved power savings and had a lower continuous mode power consumption penalty than the clock gating technique. There was a 92% reduction in idle

mode power consumption and a 5% increase for continuous mode operation. (figure 4). Thus while there is some power overhead associated with the additional reset logic, applications that have extended periods of idle mode operation will have substantial overall power savings.

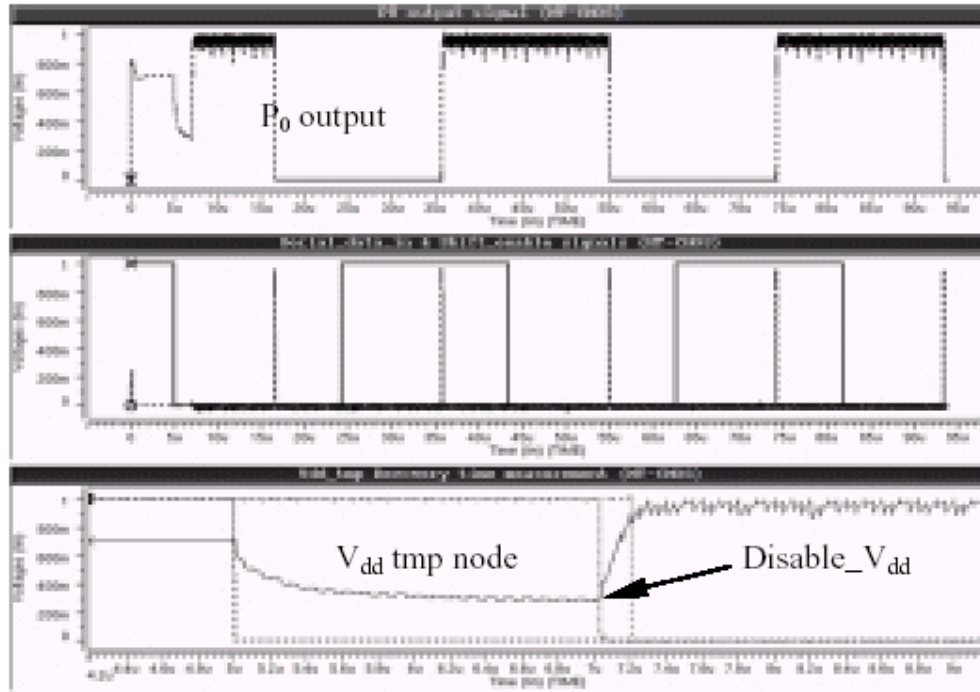


Figure 7. MT-CMOS Output waveforms.

3.4 Complete power shutdown

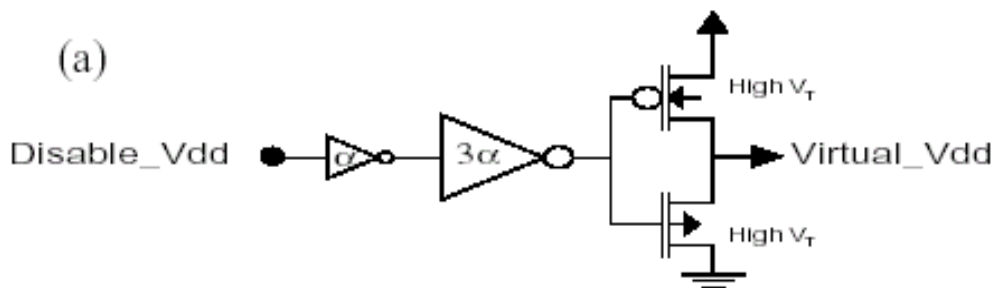
Alternatively, the disabled V_{dd} signal can be sent to a 1volt DC-DC converter. (Figure 8a) to implement a complete power shut down. The circuit in figure 8a is simply a chain of inverters where the last inverter consists of large width and high V_t PMOS and NMOS devices that acts as a switch. The first two inverters are buffers to drive the large width devices. Using this technique during the idle mode the leakage power for the UART is effectively

0. each wake up event takes approximately 100nseconds (figure 8b) along with power consumption of 0.12mWatt.

The power shutdown technique shows approximately 91% power reduction (this also takes into account the power consumption of the DC-DC converter.) in idle mode power. If this technique is applied the converter virtually eliminates all the leakage power of the UART in the idle mode. However, since all nodes are completely discharged after a byte is acknowledged the continuous mode power consumption is significantly increased due to DC-DC converter having to recharge all nodes to initialize the system.

3.5 Data-Stream dependant shutdown

To avoid node discharging after every byte, a data – stream dependant shutdown procedure was considered. In this scenario, an independent decoder system would monitor the receiving Data-Stream for a special end-of-transmission byte.



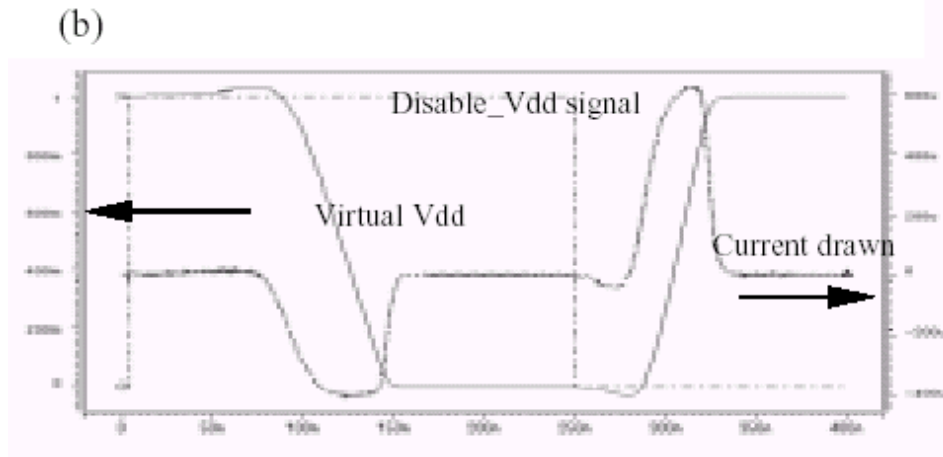


Figure 8. Vdd cutoff circuit and rise/fall times.

When this special byte is received the decoding system signals the DC-DC converter to shutdown the UART supply voltage. Consequently, the shutdown frequency is greatly reduced. However, there is large overhead associated with decoding the Data-Stream for the special end-of-transmission byte. To reduce the decoding overhead we analyzed another scheme where the number of data bits transmitted is 'n+1' rather than 'n'. The extra bit when set to 1, will indicate end of a Data-Stream and a disabled V_{dd} signal will be generated.

The logic overhead for transmitting one extra bit requires one more block in the shift register, which has very low activity. To maintain performance (throughput) in the case of 10MHz clock, frequency of 10.7MHz is required. This linearly increases the continuous mode power consumption by 0.36microWatts. (9.2 %). This power is not significant compared to the power consumption for V_{dd} cut-off after each byte. It is apparent that in the continuous mode of operation, the method of transmitting an extra bit to shutdown the system after a Data-Stream is more power efficient than the regular implementation of V_{dd} cut-off after every byte. Unfortunately during

the idle mode of operation, the DC-DC converter's idle power (due to the leakage current in large width switches) is larger than the idle mode power in the unmodified (no DC-DC converter) circuit.

3.6 Low transition ASCII (LT-ASCII)

Once the logic circuitry was scaled to operate at one volt power supply it was apparent that the I/O unit with a 5volts rail – to – rail swing dominates the overall continuous mode power of the UART (figure 3). Lowering the voltage swing is not an obvious solution since serial I/O signal usually need to go off chip through short wires. To lower output pad power consumption a special ASCII – compatible data encoding scheme, Low transition ASCII (LT-ASCII), is proposed which reduces I/O switching activity.

For a given UART application (in a keyboard for example), certain character values will occur more frequently than others. A larger number of serial bit transmissions for certain input character bytes will result in a higher power consumption for the UART. By encoding the most commonly used characters (e.g. a,e,i,n,o,r,s,t) into a 10000000 to 11111111 byte re-assignment, the characters can be transmitted from the UART with fewer transitions in the I/O pad. In the encoding byte, the first bit (which is not used in normal ASCII coding) is changed to a 1 to specify to the system that the byte is encoded. For example, 'e' character binary representation (01100101) requires six 0 to 1 / 1 to 0 transitions including the two 0 protocol bits at either end. Encoding 'e' to 11111110 results in only two transitions. Figure 9 shows character power consumption before and after data encoding. Character probabilities were extrapolated from a 15000 word documents sample. The use of data encoding decreased I/O pad path by 38%.

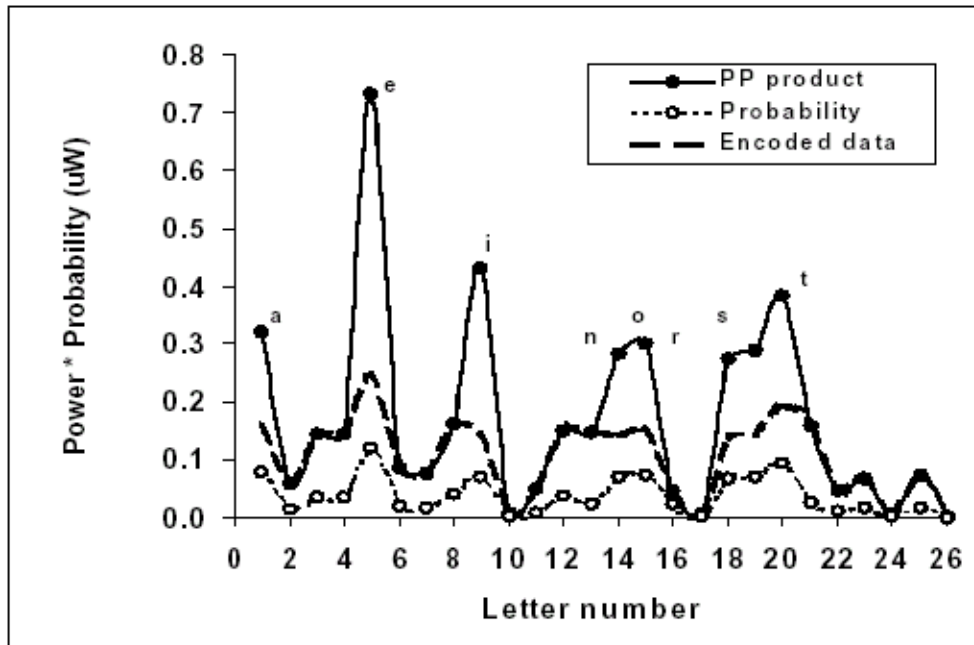


Figure 9. I/O Power*Probabilty product vs. ASCII character.

4. CONCLUSION

Several low power configurations were examined to minimize the UART power consumption. By both MT – CMOS and clock gating techniques reduced idle mode power significantly; there was an increase in continuous mode power consumption due to the logic overhead. With optimization, the MT – CMOS design had a logic power overhead of only 5% compared to 33% overhead cost for clock gating. In other technique complete power shutdown allows zero leakage current for the UART in idle mode. However, the idle mode power for the DC-DC converter circuit was seen to be larger than the UART idle power with MT – CMOS. Therefore, it is more power efficient to implement MT – CMOS rather than complete power shutdown. Finally, with the proposed Low transition ASCII (LT-ASCII) encoding scheme, fewer transitions can be realized on the output pad. The optimal system configuration that resulted in minimal power consumption was low voltage operation with MT – CMOS and the Low transition ASCII (LT-ASCII) encoding scheme. Power saving researches is summarized in table 1.

Table 1. Summary of Power Savings.

| Frequency = 10 MHz | Power savings Cont. mode | Power savings Idle mode | Power savings Output pad |
|---------------------|-----------------------------|----------------------------|-----------------------------|
| 1V operation | 97% | 97% | 0% |
| With clock gating | -33% | 78% | 0% |
| With MT-CMOS | -5% | 92% | 0% |
| With ASCII encoding | 0% | 0% | 38% |

6. REFERENCES

- a) A.P. Chandrakasan, S. Sheng and R.W. Brodersen, "Low Power CMOS Digital Design". IEEE J. Solid-State circuits, Vol.27, no.4, Apr.1992,pp.473-484.
- b) S. Mutoh, T.Douseki, Y.Matsuya, T. Aoki, S.Shigematsu and J.Yamada, "1-V power supply High Speed Digital Circuit Technology with Multi Threshold – Voltage CMOS" IEEE J. Solid-State circuits, August1995,pp.847-854.
- c) D.Liu, and C.Svensson, "Trading Speed for Low Power by Choice of Supply and Threshold Voltages". IEEE J. Solid-State circuits, Vol.28, no.1, Jan.1993, pp.10-17.
- d) P. Horowitz, W. Hill, "The Art of Microelectronics", Second edition, pp. 711-812.
- e) A. Clements, "Microprocessor System Design, 68000 Hardware, Software, and Interfacing", Third edition. Pp. 701-744.

ABSTRACT

In this paper, several low power techniques, including low transition ASCII (LT-ASCII), that are particularly effective for the universal asynchronous receiver / transmitter (UART) circuits are proposed. Low power techniques, such as voltage scaling, clock gating, MT-CMOS and complete power shut down, are investigated to minimize the UART logic power in both continuous and idle modes of operation. The overheads circuitry required for each technique and penalties associated with their own power consumption are discussed. Finally, the proposed LT-ASCII is an ASCII compatible protocol which contributes to significant power savings by minimizing activity factor of the UARTs I/O pad.

ACKNOWLEDGEMENT

I extend my sincere thanks to **Prof. P.V.Abdul Hameed**, Head of the Department for providing me with the guidance and facilities for the Seminar.

I express my sincere gratitude to Seminar coordinator **Mr. Berly C.J**, Staff in charge, for their cooperation and guidance for preparing and presenting this seminar.

I also extend my sincere thanks to all other faculty members of Electronics and Communication Department and my friends for their support and encouragement.

MOHAMED ABDUL KARIM.N

CONTENTS

1. INTRODUCTION
2. UART DESIGN AND PROTOCOL
3. RESULTS AND DISCUSSION
 - Voltage-Scaling
 - Clock-Gating
 - Multi-Threshold CMOS (MT-CMOS)
 - Complete Power Shutdown
 - Data –Stream Dependent Shutdown
 - Low Transition ASCII (LT-ASCII)
4. CONCLUSION
5. REFERENCES