Whitepaper 3: De-Clocking vs MCU Standby for Low Power Design

Abstract

The need for reducing power is ever increasing with demands for a greener environment, and as such battery powered designs are coming under increasing pressure to reduce power and make the battery last longer.

Many applications such as heating controllers, meters and remote sensors etc. are required to work in a repetitive way, where the MCU can idle for large periods of the time waking periodically to make measurements, adjustments or communication, while in other applications increased levels of performance are required which is not an ideal combination for saving power. This does raise a question for these applications:

"Can a modern 32-bit MCU providing higher performance still have sufficiently low power operation when compared to a similar generation 16-bit product?"

Introduction

This paper, the third in a series of four whitepapers examining the various low power design techniques based on specific examples using the Renesas 16-bit RL78 (RL78/L12) and the 32-bit RX100 (RX111) as outlined in whitepaper 1 (powering your system from a lemon!). While the results may vary slightly the principles in this paper apply equally to the other RL78 and RX microcontroller families.

In this paper we examine the effects on average power consumption of running the MCU at lower clock frequencies ("de-clocking") compared to using the MCU standby (sleep) modes during periods of inactivity. Note that the maximum frequency used in the calculated examples below is 8 MHz representing a "medium" performance level, whereas the maximum clock speeds of the RL78/L12 and RX111 is 24 MHz and 32 MHz respectively. It would be relatively straight forward to calculate the average power of these products if it was necessary to operate at full speed.

Sleep Modes vs De-Clocking to save power

Whitepapers 1 and 2 have analysed the different sleep (standby) modes and some of the effects of lowering the system clock frequencies, but the question analysed in this paper is

"Is it better to use standby operation or run the MCU permanently at a slower clock frequency"?

First let's consider what is meant by the term "de-clocking". To some extent the question has been already answered above and perhaps is considered relatively obvious, however two possible scenarios are considered for the purposes of this paper:

1. Running the MCU at a low fixed clock frequency but not changing frequency nor using standby during system idle times.
2. Running the MCU at a low frequency but using a slower clock during system idle times, but not using any standby modes.

By way of example this paper will consider a basic scenario where the MCU runs a foreground process and an idle period. The ratio of foreground to idle time is ~1% over a repetitive period of 1 second as shown in figure 1 below. The foreground time of 1% (1 ms) assumes a main clock frequency of 1 MHz, with the foreground times for the other clock frequencies scaled accordingly as the total period time is fixed at 1s regardless of frequency.

The results are then compared against using standby modes such as HALT/SLEEP and STOP/SW STANDBY as provided by the RL78 and RX100 families.

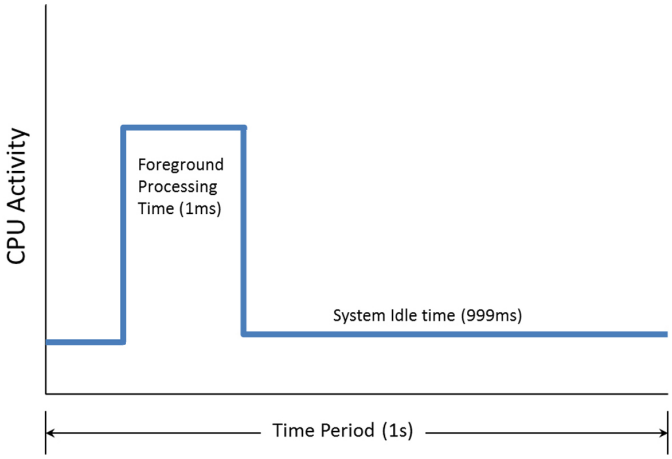

Figure 1 – System Operation Example

The average current calculations used in the tables below are based on the equation from white paper 2

$$(I_{ave} = ((AC1 * AT1) + (IC * IT)) / P),$$

where the foreground processing power consumption is set by Active Current 1 and Active Time 1 (AC1 & AT1) with the Idle Time and Idle Current are set by IC and IT. Normal peripheral operation is assumed during foreground processing and no peripheral operation during idle times with the exception of the RTC.

A nominal 3 Volt supply is used for all currents and applies to both RL78/L12 and RX111 calculations.

RL78 Average Current Calculations

The figures calculated in table 1 are based on using a fixed clock frequency with no change during idle times.

Table 1 – RL78 (fixed clock frequency)					
Clock Frequency	Foreground Time	Foreground Current	Idle Time	Idle Current	Average Current
8 MHz	125 µs	1.8 mA	999.87 ms	1.2 mA	1.20 mA
4 MHz	250 µs	1.7 mA	999.75 ms	1.2 mA	1.20 mA
1 MHz	1 ms	1.19 mA	999 ms	0.9 mA	0.90 mA
32 kHz	32 ms	5.0 µA	968 ms	3.7 µA	3.74 µA

The figures calculated in table 2 below are based on using two frequencies, a high speed clock for foreground processing and the sub-clock (32 KHz) during the idle times.

Table 2 – RL78 (HS clock for foreground, LS clock during idle time)					
Clock Frequency	Foreground Time	Foreground Current	Idle Time	Idle Current (32 KHz)	Average Current
8 MHz	125 µs	1.8 mA	999.87 ms	3.7 µA	3.92 µA
4 MHz	250 µs	1.7 mA	999.75 ms	3.7 µA	4.12 µA
1 MHz	1 ms	1.19 mA	999 ms	3.7 µA	4.89 µA

From the results in the table 1 above, operating from a single frequency does not provide any significant power reduction except when using low clock speeds.
Not surprisingly using the low speed sub-clock clock during the large idle period (table 2) makes a big difference when using the higher clock frequencies during the foreground operation, but again running at the fixed sub-clock still provides a lower average current.
However if we factor in the use of the standby operation during the idle period into the calculations, (table 3 below) we can see a big change to in the average current figures.

Table 3 – RL78 (HS clock for foreground, Standby during idle time)							
Clock Frequency	Foreground Time	Foreground Current	Idle Time	HALT Current (32 KHz)	STOP Current (32 KHz)	Average Current (HALT)	Average Current (STOP)
8 MHz	125 µs	1.8 mA	999.87 ms	0.56 µA	0.23 µA	0.785 µA	0.455 µA
4 MHz	250 µs	1.7 mA	999.75 ms	0.56 µA	0.23 µA	0.985 µA	0.655 µA
1 MHz	1 ms	1.19 mA	999 ms	0.56 µA	0.23 µA	1.749 µA	1.420 µA
32 kHz	32 ms	5.0 µA	968 ms	0.56 µA	0.23 µA	0.702 µA	0.383 µA

From the results in table 3, we now see that for all clock frequencies utilising HALT or STOP provides dramatic power savings where again the low speed sub-clock still offering the best overall average current consumption.
Although interestingly, using the higher foreground clock and the low speed in standby produces results very close to running the system permanently at the sub clock frequency (32 KHz).

RX111 Examples

Now let's examine the same scenarios but with the more powerful 32-bit MCU, the RX111.

Table 4 – RX111 (Fixed clock frequency)					
Clock Frequency	Foreground Time	Foreground Current	Idle Time	Idle Current	Average Current
8 MHz	125 µs	3.5 mA	999.87 ms	1.3 mA	1.30 mA
4 MHz	250 µs	2.35 mA	999.75 ms	1 mA	1.00 µA
1 MHz	1 ms	1.2 mA	999 ms	750 µA	0.75 µA
32 kHz	32 ms	11.5 mA	968 ms	4.0 µA	4.24 µA

The figures in table 4 are based on using a fixed frequency with no change during the idle time.

Table 5 – RX111 (HS clock for foreground, LS clock during idle time)					
Clock Frequency	Foreground Time	Foreground Current	Idle Time	Idle Current (32 KHz)	Average Current
8 MHz	125 µs	3.5 mA	999.87 ms	4.0 µA	4.44 mA
4 MHz	250 µs	2.35 mA	999.75 ms	4.0 µA	4.59 µA
1 MHz	1 ms	1.2 mA	999 ms	4.0 µA	5.20 µA

The figures in table 5 are based on using two frequencies, a high speed clock for foreground processing and the 32 KHz sub-clock during the idle time.

The RX111 results in tables 4 and 5 above show a very similar picture to that of the RL78 where operating from a single frequency provides little power reduction, but using the low speed sub-clock makes a big difference.

However if we again factor in the use of standby operation during the idle periods (table 6) into the calculations then we see similar levels of power reduction and current when using the higher speed main clock and low speed sub clock combination, as shown for the RL78.

Table 6 – RX111 (HS clock for foreground, Standby during idle time)							
Clock Frequency	Foreground Time	Foreground Current	Idle Time	Deep Sleep Current (32 KHz)	SW Standby Current (32 KHz)	Average Current (Deep Sleep)	Average Current (SW Standby)
8 MHz	125 µs	3.5 mA	999.87 ms	1.8 µA	0.35 µA	2.237 µA	0.787 µA
4 MHz	250 µs	2.35 mA	999.75 ms	1.8 µA	0.35 µA	2.387 µA	0.937 µA
1 MHz	1 ms	1.2 mA	999 ms	1.8 µA	0.35 µA	2.998 µA	1.550 µA
32 kHz	32 ms	11.5 µA	968 ms	1.8 µA	0.35 µA	2.110 µA	0.707 µA

Conclusion

The theme of this paper was to consider the effects on power consumption when applications operate on repetitive cycle basis and the different results of operating the MCU at a lower fixed clock frequency, using a low speed "sub clock" during the system idle times and lastly to compare these results against the use of the MCU standby modes during the idle periods.

While a continuous fixed clock frequency unsurprisingly showed little usable power reduction for both of our sample families, changing the operating clock to use a sub-clock showed some good improvements for both MCU's and in some applications could possibly be used as a solution.

However it is clear that for both MCU families the answer to the question

"Is it better to use standby operation or run the MCU permanently at a slower clock frequency?"

is that using standby modes during idle times provides the lowest average current and confirms that this is still the best design approach wherever possible to make the battery last the longest.

Interestingly the combination of using a higher main clock frequency during the foreground processing together with the sub clock and standby operation during the idle periods shows very similar results to that of running the design at low speed. This allows a choice depending on the needs of the applications such that where the processing needs are low such as the "lemon" demonstration concept (whitepaper 1) the lowest current consumption is achieved

with everything operating from a low speed clock and using standby. When foreground processing demands are higher, operating with a faster main clock (i.e. 8 MHz) together with standby produces a very small increase in current consumption, such that in our calculated examples above the increase was only 72 nA for the RL78/L12 and 80 nA for the RX111.

The other question posed at the start of thiswhite paper was *"Can a modern 32-bit MCU providing higher performance still have sufficiently low power operation when compared to a similar generation 16-bit product?"*

Well the answer is yes, that when employing standby operation a modern 32-bit MCU family such as the RX100 (RX111) can produce very low average currents close to that of the 16-bit RL78 (RL78/L12), combining low power with higher performance and functionality when applications demand more from the design.

For further information it is recommended to read the other whitepapers in this series (highlighted below) and please visit the Renesas design resources centre.

Whitepaper 1: Lemon Powered Design

An example of what can be achieved with the right product and modes of operation

Whitepaper 2: The Rules of Low Power MCU Design

Analysis of many techniques to provide the lowest power consumption possible

Whitepaper 4: Maximise Your Battery Life

Analysis of systems that are designed to spend long periods in standby operation

Written by: David Parsons - Consultant to Renesas Electronics (Europe) GmbH
David can be contacted at DCP Electronics and Software Services.

Document No. R01PF0080ED0200