1 Test clock settings

Clock Settings [MHz]	2	12	33	48	66
PLL Frequency [MHz]	48	48	66	96	132
PLLMUL	3	3	10	15	10
PLLDIV	0	0	1	1	1
PLL_FREQ [MHz]	48	48	66	96	132
PLL_DIV2	1	1	1	1	0
PBADIV	1	1	1	1	1
PBASEL	0	0	0	0	0
PBBDIV	1	1	1	1	1
PBBSEL	0	0	0	0	0
HSBDIV	1	1	1	1	1
HSBSEL (CPUSEL)	3	1	0	0	0
Works as expected?	Yes	Yes	Yes	Yes	
Approximate blinking freq. [Hz]	1/4	1	3	4	5
Measured power consumption [mW]	515	545	620	680	730

Conclusion
Power consumtion and blink frequency seems to be increasing proportionally to the clock frequency.

2 Using support functions

a. The Slow Clock is called $RCSYS$ in the header file. It is called from the function $pcl\_configure\_clocks\_rcsys(pc\_freq\_param\_t * param)$.
b. The struct has definition

```
typedef struct
{
  //! Main clock source selection (input argument).
  pcl_mainclk_t main_clk_src;

  //! Target CPU frequency (input/output argument).
  unsigned long cpu_f;

  //! Target PBA frequency (input/output argument).
  unsigned long pba_f;

  //! Target PBB frequency (input/output argument).
  unsigned long pbb_f;

  //! Target PBC frequency (input/output argument).
  unsigned long pbc_f;

  //! Oscillator 0's external crystal(or external clock)
      frequency (board dependant) (input argument).
  unsigned long osc0_f;

  //! Oscillator 0's external crystal(or external clock)
      startup time: AVR32_PM_OSCCTRL0_STARTUP_x_RCOSC (
      input argument).
  unsigned long osc0_startup;

  //! DFLL target frequency (input/output argument) (NOTE
      : the bigger, the most stable the frequency)
  unsigned long dfll_f;

  //! Other parameters that might be necessary depending
      on the device (implementation-dependent).
  // For the UC3L DFLL setup, this parameter should be
      pointing to a structure of
  // type (scif_gclk_opt_t *).
  void *pextra_params;
} pcl_freq_param_t;
```