



Practical Power Testing



A Reference Design for the Spartan™-3A FPGA Starter Kit

PicoBlaze™

Ken Chapman
Xilinx Ltd

Rev1 – 29th February 2008



Limitations

Limited Warranty and Disclaimer. These designs are provided to you “as is”. Xilinx and its licensors make and you receive no warranties or conditions, express, implied, statutory or otherwise, and Xilinx specifically disclaims any implied warranties of merchantability, non-infringement, or fitness for a particular purpose. Xilinx does not warrant that the functions contained in these designs will meet your requirements, or that the operation of these designs will be uninterrupted or error free, or that defects in the Designs will be corrected. Furthermore, Xilinx does not warrant or make any representations regarding use or the results of the use of the designs in terms of correctness, accuracy, reliability, or otherwise.

Limitation of Liability. In no event will Xilinx or its licensors be liable for any loss of data, lost profits, cost or procurement of substitute goods or services, or for any special, incidental, consequential, or indirect damages arising from the use or operation of the designs or accompanying documentation, however caused and on any theory of liability. This limitation will apply even if Xilinx has been advised of the possibility of such damage. This limitation shall apply not-withstanding the failure of the essential purpose of any limited remedies herein.

This design module is **not** supported by general Xilinx Technical support as an official Xilinx Product. Please refer any issues initially to the provider of the module.

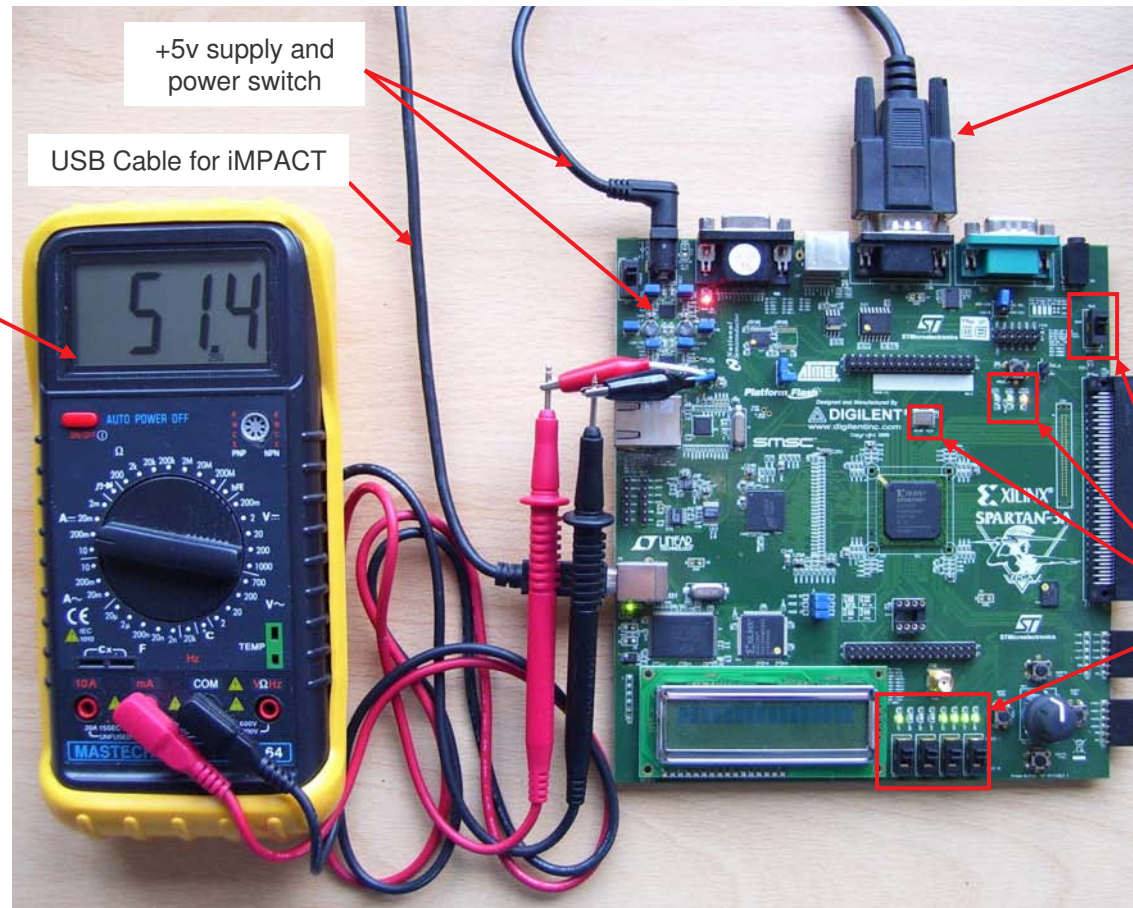
Any problems or items felt of value in the continued improvement of this reference design would be gratefully received by the author.

Ken Chapman
Senior Staff Engineer
Spartan Applications Specialist
email: chapman@xilinx.com

Introduction

This reference design has been created to enable you to conduct experiments and measurements to determine the actual power consumption of the XC3S700A device on your Spartan-3A Starter Kit. As well as connecting an ammeter to measure supply currents, it will be necessary for you to study this document adequately to understand the ways in which you are able to control the design and make sense of the measurements you take.

Although this design is provided for the 'Starter Kit' it is not recommended as the first design to use with the board, but rather it is a design for those specifically looking to investigate and understand power dissipation. As well as enabling you to directly observe the relatively low power demands of the Spartan-3A device, it is hoped that this reference design will also equip you with knowledge which will assist you in implementing designs which consume less power in the future.



+5v supply and power switch

USB Cable for iMPACT

RS232 Serial Cable connects to PC running HyperTerminal (or similar) and needs to be a male to female straight through cable (critically pin2-pin2, pin3-pin3 and pin5-pin5).

9600 Baud
8-bit
1 stop bit
No parity
Flow control: None

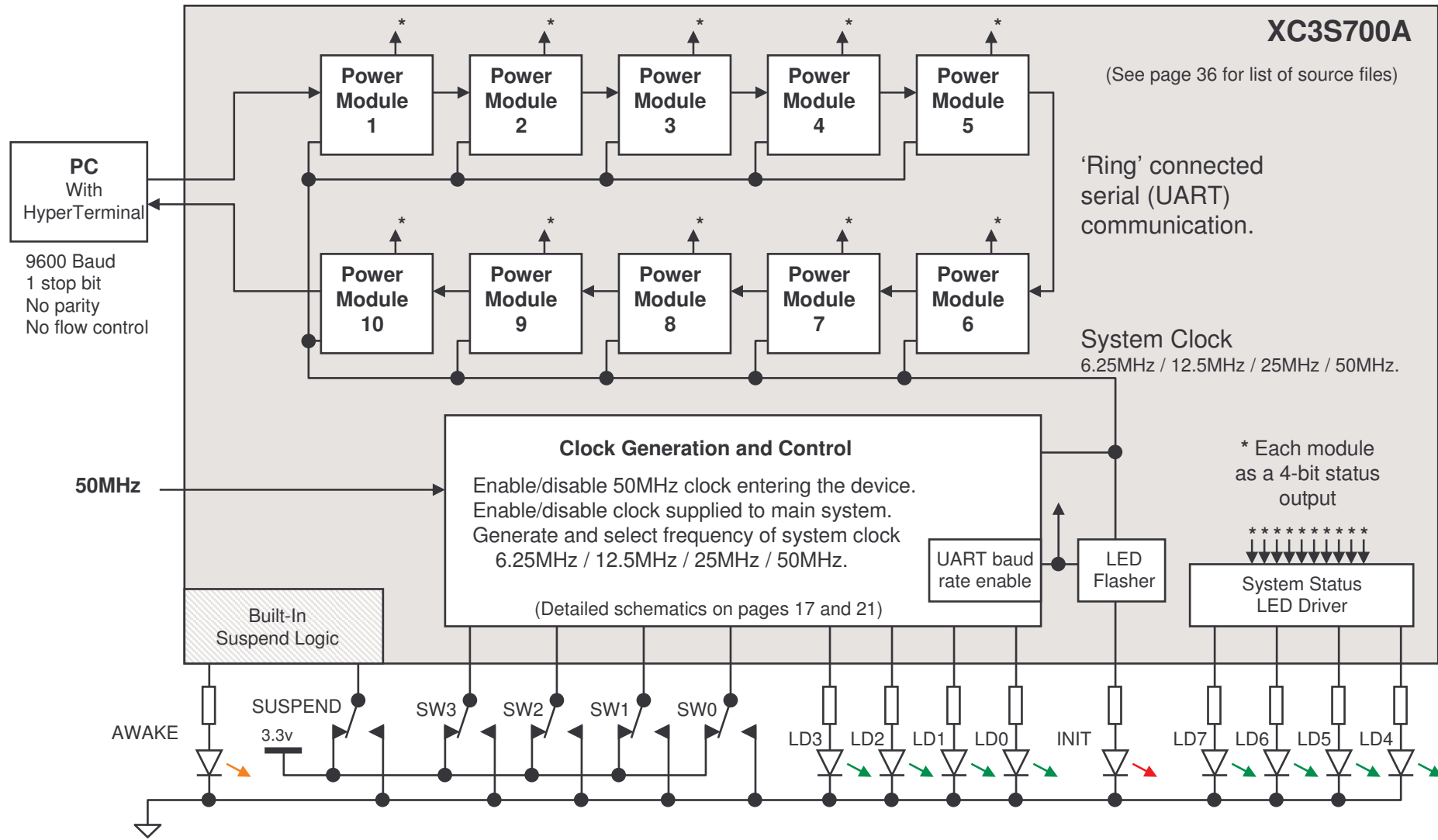
The reference design also makes use of the LEDs, slide switches and 50MHz oscillator.

Connect an ammeter to measure current demand of the 1.2v V_{CCINT} and 3.3v V_{CCAUX} power rails (See page 7 for details of connections).

The design supplied will result in current measurements between 1mA and ~110mA so choose an appropriate scale on your meter.

Design Overview

The reference design contains 10 identical modules whose sole purpose is to cause power to be dissipated in a predictable manner controlled by simple commands entered at the PC terminal. Switches directly control the suspend mode and clocks with LEDs confirming the status of the device and design.



Power Module Overview

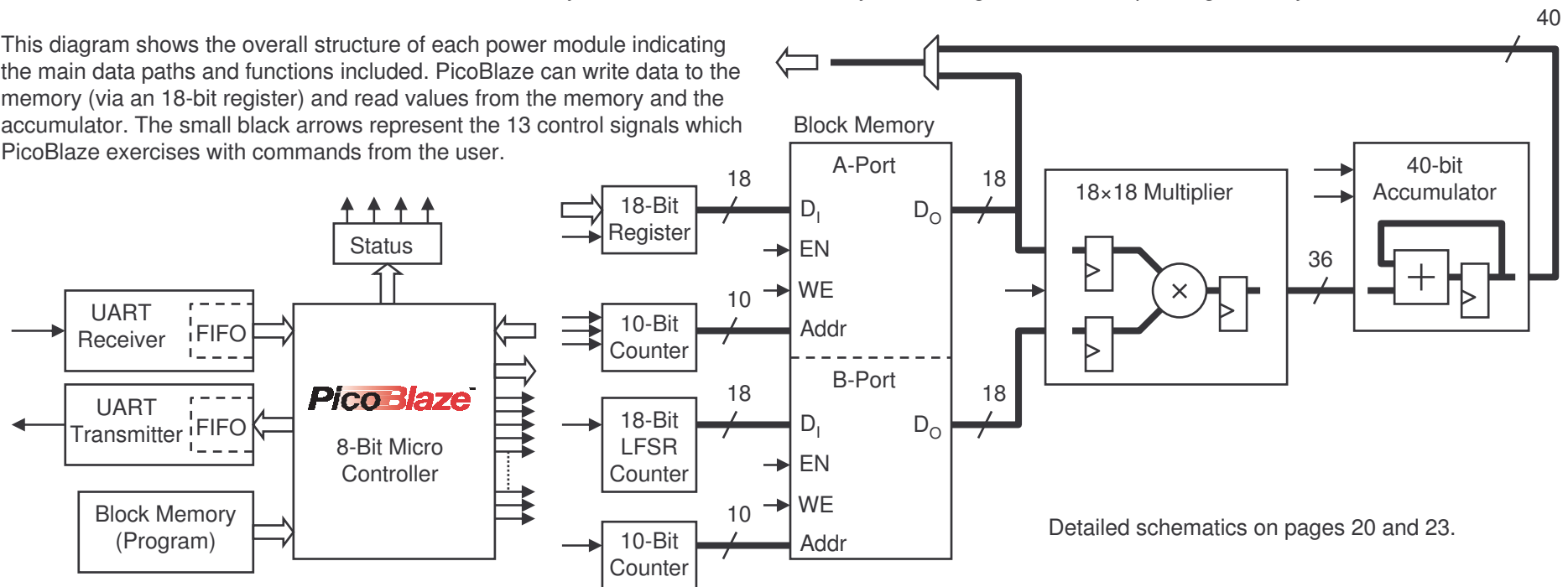
The 'power_testing' design includes a 'power module' which is replicated 10 times. The replication is simply to magnify (or amplify) the power consumption and enable more accurate supply current measurements to be made. **Remember to divide all current measurements by 10 when calculating the power consumption associated with any element contained in the power module.**

Each power module contains an 8-bit PicoBlaze processor which in itself represents a good balance of logic resources including a Block Memory in which the program is stored and 96 'slices' of logic implementing general logic functions such as counters, multiplexers, add/subtract and decoding as well as the highly efficient distributed memory used to implement 16 registers, 64-bytes of scratch pad memory and the call/return stack.

PicoBlaze is connected to the rest of the system using UART receiver and transmitter macros with integral 16-byte FIFO buffers. These enable you to control both PicoBlaze and the further logic functions connected to it in a totally predictable way in order to make precise current and power measurements.

The additional functions are based on a multiply and accumulate structure which is fed with data values from a second Block Memory. This memory in turn has address counters. The memory can be loaded with data patterns by PicoBlaze or written at full clock rate with pseudo random data from an LFSR counter. PicoBlaze can read back values from the memory and the accumulator to verify that the logic structure is operating correctly.

This diagram shows the overall structure of each power module indicating the main data paths and functions included. PicoBlaze can write data to the memory (via an 18-bit register) and read values from the memory and the accumulator. The small black arrows represent the 13 control signals which PicoBlaze exercises with commands from the user.



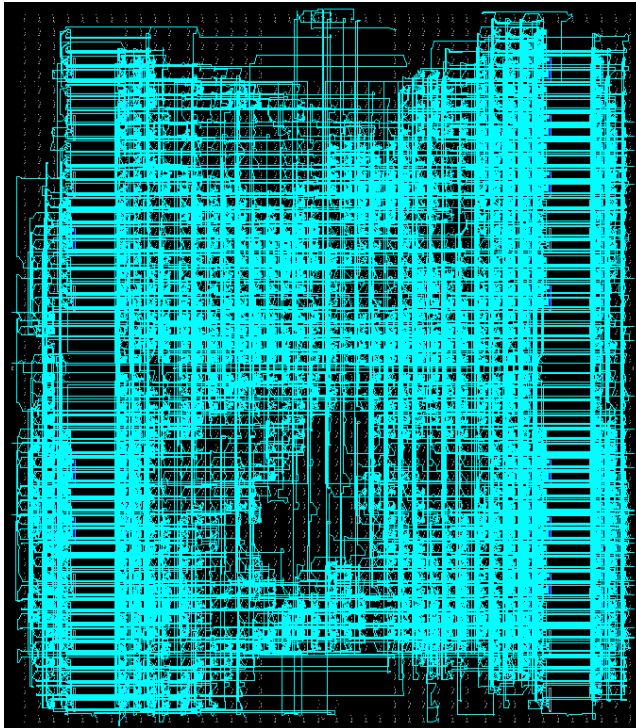
Design Size

This reference design occupies 39% of the logic slices, 50% of the multipliers and 100% of the Block Memories. The object of the design is to facilitate current supply measurements to determine power dissipation of different functions so the actual device utilisation is not particularly significant. What is significant is that the utilisation of multipliers and Block Memories is high because they are only available in relatively small numbers compared with the thousands of logic slices and an adequate number should be used to enable supply current measurements to be made with reasonable accuracy.

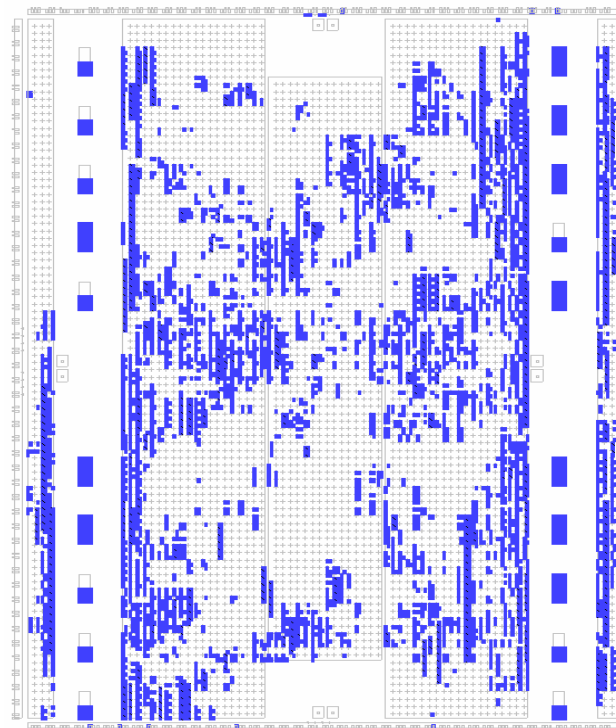
MAP report

Number of occupied Slices:	2,337 out of	5,888	39%
Number of RAMB16BWEs:	20 out of	20	100%
Number of MULT18X18SIOs:	10 out of	20	50%

FPGA Editor view

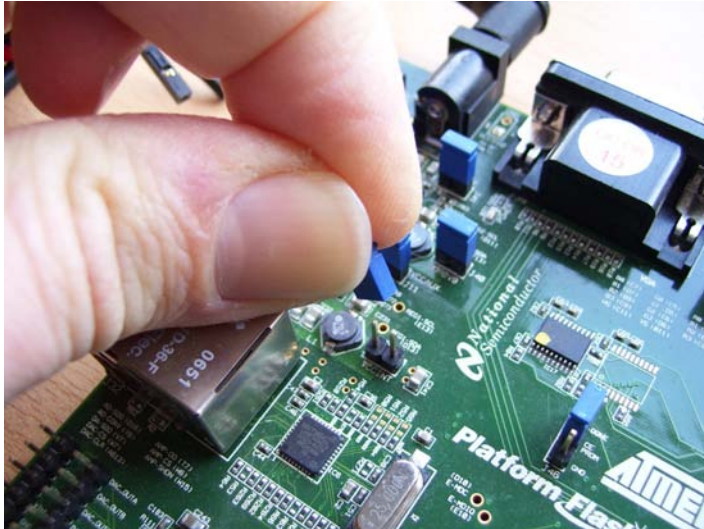


Floorplanner view



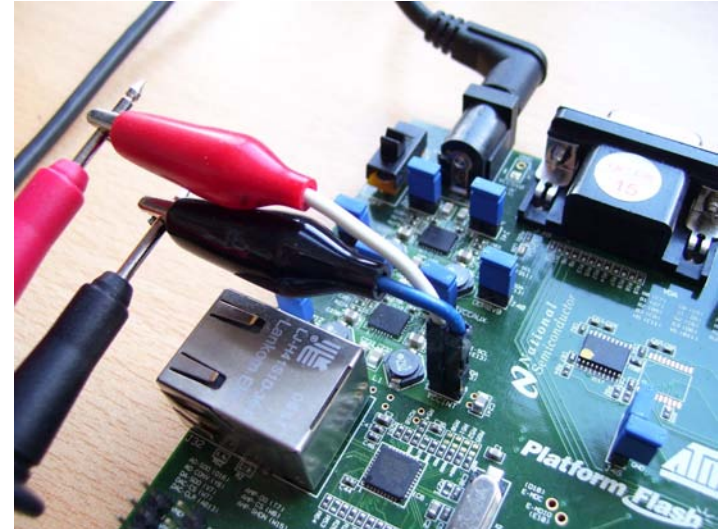
Connecting a Meter

The Spartan-3A Starter Kit is provided with removable jumpers in both the 1.2v V_{CCINT} and 3.3v V_{CCAUX} power rails. Simply remove the desired jumper and connect your ammeter in its place. It is probably wise to turn off the power supply whilst making your connections!



Illustrated here, the V_{CCINT} jumper is removed and the ammeter connected using some clips. The positive lead is on the left (identified by the red clip in this picture).

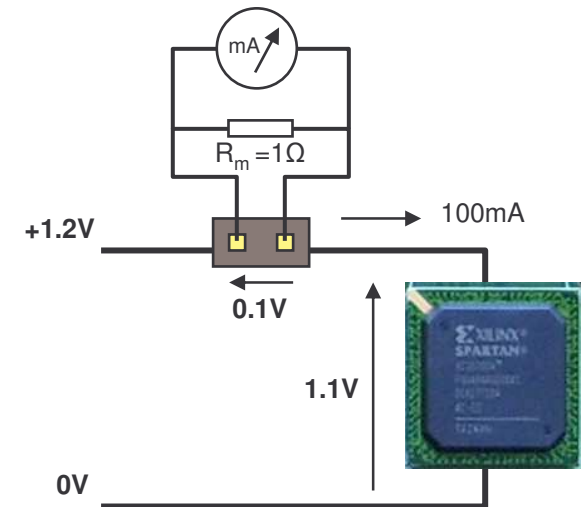
The V_{CCAUX} jumper is slightly up to the left and can be connected to in the same way.



BEWARE the Measurement!

Just a polite reminder of the fact that it is impossible to measure any system without effecting the very system which you are attempting to measure! Besides the absolute accuracy of an instrument, any ammeter will have an internal 'shunt' resistance. As current flows through the meter there will be a voltage drop across the meter which in turn will lower the voltage applied to the Spartan-3A device which in turn will cause the Spartan-3A to draw less current than it would at full voltage. In a 12v world, or even +5v TTL circuits, the voltage drop associated with a meter could generally be ignored, but with a V_{CCINT} of just 1.2v, even small voltage drops become significant. This diagram indicates how an ammeter with an internal resistance of just 1Ω is enough to cause the Spartan-3A to operate below its recommended level of 1.14v when 100mA is being drawn from the 1.2v supply.

Note – The DVM used by the author and shown in this document was not a fully calibrated instrument but the author has no reason to believe that the measurements shown are inaccurate other than those caused by measurement errors of the type described here.



Serial Terminal Setup

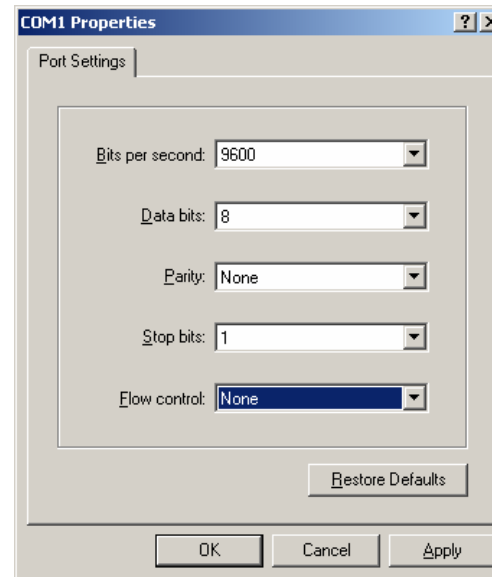
An RS232 serial link is used to communicate with the 'power_testing' reference design and to control the various features such that supply current measurements can be made. Any simple terminal program can be used, but HyperTerminal is adequate for the task and available on most PCs.

A new HyperTerminal session can be started and configured as shown in the following steps. These also indicate the communication settings and protocol required by an alternative terminal utility.

- 1) Begin a new session with a suitable name.
HyperTerminal can typically be located on your PC at
Programs -> Accessories -> Communications -> HyperTerminal.



- 2) Select the appropriate COM port (typically COM1 or COM2) from the list of options. Don't worry if you are not sure exactly which one is correct for your PC because you can change it later.



- 3) Set serial port settings.

Bits per second : 9600
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None

Go to next page to complete set up...

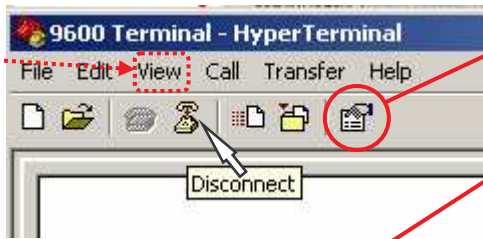


HyperTerminal Setup

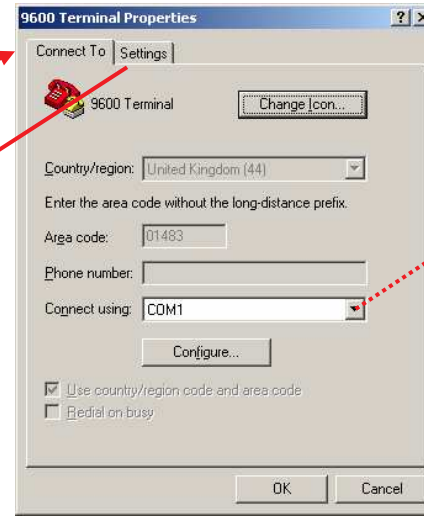
Although steps 1, 2 and 3 will actually create a Hyper terminal session, there are few other protocol settings which need to be set or verified for the PicoBlaze design to work as expected.

Optional step.....
Set Font to
Courier New,
Regular, 10

4 - Disconnect

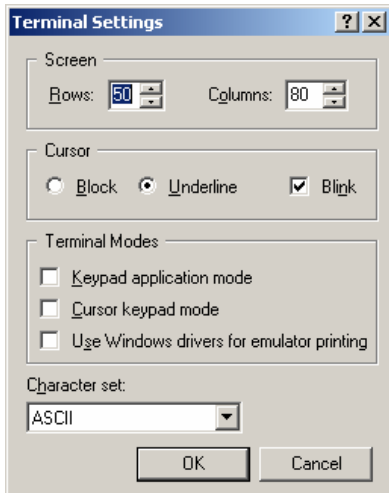


5 - Open the properties dialogue

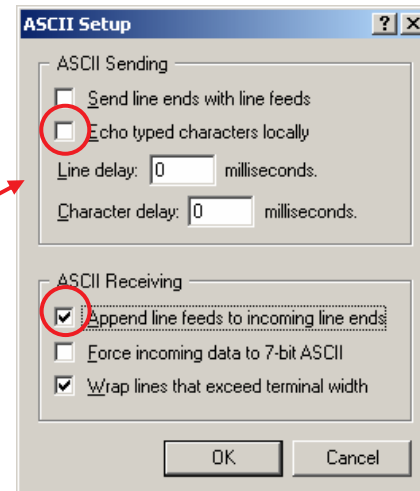
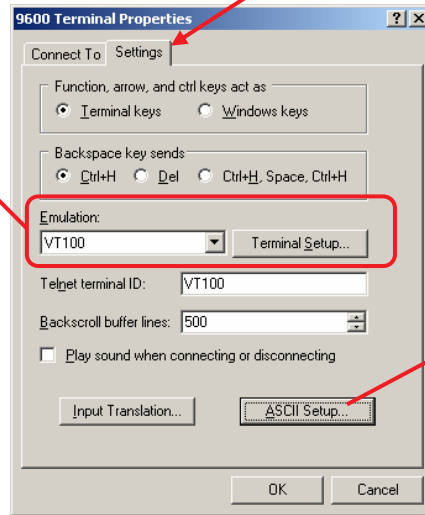


To select a different
COM port and change
settings (if not correct).

7 - Select VT100**
then 'Terminal Setup'.
Set 'Rows' to 50.



6 - Open Settings



8 - Open ASCII Setup

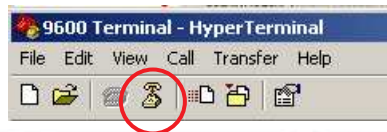
Ensure boxes are filled in as shown.

The design will echo characters that you type so you do not need the 'Echo typed characters locally' option.

The design transmits carriage return characters (OD_{HEX}) to indicate end of line so you do need the 'Append line feeds to incoming line ends' option to be enabled.

Note - You will probably need to stretch the main screen to fit the new size and font.

9 - 'OK' the boxes to get back to main screen and then Connect.



**Hint – You may have to use HyperTerminal once and then return to the 'Terminal Settings' window again to reveal the 'Screen' options.

Configure Spartan-3A

With your board and PC all ready to go it is time to configure the Spartan-3A with the 'power_testing' design.

- 1) Connect the power, USB cable and RS232 serial cable to the board as shown on page 3.
- 2) Connect an ammeter as shown on page 5. (You may defer this step initially but you will need one soon!)
- 3) Open HyperTerminal set up as described on pages 6 & 7.
- 4) Place the SW4 switch (upper-right side of board) to the RUN position.
- 5) Move all user slide switches SW0, SW1, SW2 and SW3 in the 'LOGIC 1' position.
- 6) Unzip all the files provided into a working directory.
- 7) Double click on the file '**install_power_testing.bat**'.

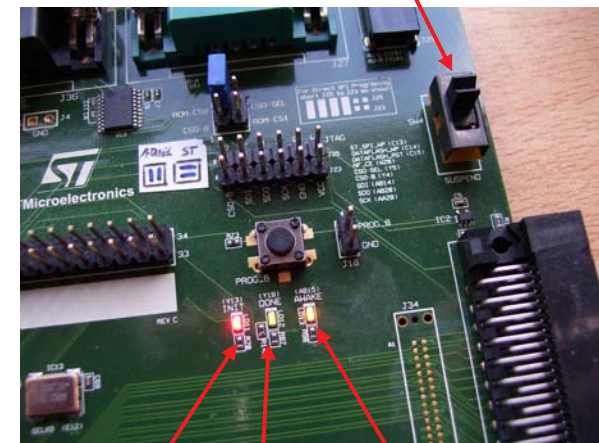
Note that you must have the Xilinx ISE tools installed on your PC

This batch file should open a DOS window and run iMPACT in batch mode to configure the Spartan XC3S700A device with the configuration 'power_testing.bit' file provided. Configuration should be confirmed by:-

- The DONE LED illuminating.
- The AWAKE LED illuminating.
- User LEDs LD0, LD1, LD2 LD3 and LD7 illuminating.
- The INIT LED flashing at one second intervals.
- Your HyperTerminal session displaying a design name, version number and simple menu as shown on the next page.

If one or more of the these points do not happen them please take time to double check all of the connections and setting described and try again.

SW4 switch in RUN position.



INIT DONE AWAKE

Recommendation

Due to the interruption to power supplies associated with connecting or moving the ammeter, it is highly recommended that the reference design be programmed into one of the on board FLASH memories. The 'power_testing_v15.mcs' file is provided for this purpose and is ideal for programming into the XCF04S Xilinx Platform FLASH memory. It is very easy to program the Platform FLASH with the provided file using iMPACT. Set the mode jumpers and you will be instantly ready to make measurements every time you apply power.

Terminal Display

The terminal display is a combined menu of command options and an indication of operational status.

The commands are described in more detail later but at this stage we just want to make sure things are working before starting to make measurements.

Hint – Just as a quick test, try enabling all features one at a time and check that your ammeter connected in the VCCINT power rail shows an increasing level of current.

```
PicoBlaze Power Control v1.15

R - Reset
C - Calculation Test
F - Functional Test
1 - Fill BRAM 1,2,3...
2 - Fill BRAM +1,-1...
3 - Display BRAM Contents
4 - (0) A-Port BRAM Enable
5 - (0) A-Port Address Counter Enable
6 - (0) B-Port BRAM Enable
7 - (0) B-Port Address Counter Enable
8 - (0) B-Port BRAM Write Enable
P - (0) PN Generator Enable
M - (0) Multiplier Enable
A - (0) Accumulator Enable

>
```

The value enclosed by brackets indicates if the corresponding logic feature is currently enabled (1) or disabled (0).

Enter a command letter of number at the prompt (upper and lower case accepted)

Power Theory – Part 1

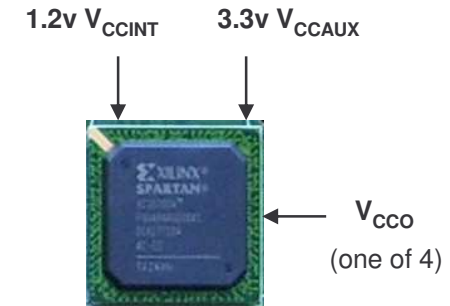
In relation to a Spartan-3A device, power dissipation is the supply voltage times the *average* supply current ($P=V \times I$). In practice, you typically measure the supply current and then calculate power dissipation. That is certainly the case with this reference design; in fact, for those concerned about battery life in portable equipment then it really is average supply current which is of most interest anyway.

The Spartan-3A device has separate supply rails for internal, auxiliary and I/O logic.

The I/O current and power is highly dependant on the external loads you connect to the device and this is not the focus of this reference design.

The auxiliary supply (V_{CCAUX}) is a separate rail and the current can be measured. On this board it is connected to 3.3v but in your own products it may be connected to 2.5v or 3.3v as convenient; for example, if you use a 3.3v supply for the I/O then you can connect the auxiliary rail to that supply also. Of course, only with a separate rail can you measure the current which is the advantage of this board. As you will see, the auxiliary supply current is relatively low, but more significantly, quite consistent.

Finally the internal supply (V_{CCINT}) is the one of most interest to us as it is the one that varies most with the operation of the design. This supply is nominally 1.2v and the current can be measured separately on the starter kit.



$$P_{INTERNAL} = (V_{CCINT} \times I_{CCINT}) + (V_{CCAUX} \times I_{CCAUX}) = (1.2 \times I_{CCINT}) + (3.3 \times I_{CCAUX})$$

If the actual internal power dissipation is important to you (typically expressed in milli-Watts), then clearly it is a simple exercise to multiply each average supply current by its associated supply voltage and combine them to deduce the total power.

Hint 1 – Although the auxiliary supply current is relatively small it is multiplied by a higher supply voltage so it is worth including.

Be wary of any manufacturers data sheets and ‘power estimators’ that do not include all supply rails in their figures.

Some devices place more burden on their auxiliary supply and it is all too easy to miss this constant drain on a battery.

Hint 2 – If power dissipation or current drain is of key concern then do try to operate V_{CCAUX} at the lower 2.5v level.

Indeed, try to operate everything including I/O at a lower voltages

Quiescent or Static Power

This is simply the current drawn by each supply rail when the device isn’t doing anything. That sounds simple but what exactly does “doing nothing” really mean? This is where it really helps to read the small print or conduct real measurements as this design and the starter kit now allows you to do yourself.

Regardless of the definition of “doing nothing”, the quiescent current will be a relatively consistent current similar to that seen with a humble resistor. Quiescent current will be the result of biasing circuits, supervisory logic and leakage and is unavoidable unless power is completely disconnected.

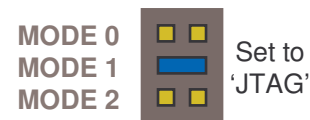
Measuring Static Power

We can now determine the quiescent or static power consumption of the Spartan-3A device. I am going to suggest several definitions of the device “doing nothing” and measure the current associated with the V_{CCINT} and V_{CCAUX} supplies. Remember that your own measurements will almost certainly be slightly different due to meter accuracy and measurement errors as well as the fundamental variations between devices and the effect of temperature. I am providing my own measurements for reference but I encourage you to make your own.

Why is Static Power Important? - Obviously low static power is a good thing in general but the point I want to make is that unless we establish the static power demands of the Spartan-3A device it will not be possible to determine the dynamic power demands of other functions later. We must therefore measure the quiescent current levels so we know what to subtract from total power measurements in future experiments.

Experiment 1 – Unconfigured Device

Method: Set the mode jumpers to JTAG as this will prevent the Spartan-3A device from attempting to configure from any FLASH memories. When power is applied to the device it will initialise (note INIT LED will blink) and then the device will “do nothing”.



Author's measurements

$$I_{CCINT} = 11.5\text{mA} \quad I_{CCAUX} = 7.9\text{mA}$$

($P_{INTERNAL} = 40\text{mW}$)

Experiment 2 – Holding PROG_B Low

Method: Regardless of the mode jumpers, press *and hold down* the PROG_B press switch such that the INIT LED remains illuminated. Note the increase in current which reflects the fact that the Spartan-3A is actually repeating the initialisation process and this means the device is doing something dynamic.

$$I_{CCINT} = 22.0\text{mA} \quad I_{CCAUX} = 9.0\text{mA}$$

($P_{INTERNAL} = 56\text{mW}$)

Hint – Clearly holding PROG_B Low is *not* a good way to minimise current. The correct way to delay configuration is to hold INIT_B Low.

If you are careful you can try this experiment using a wire link between GND and the contact to the INIT LED as power is applied or PROG_B cycled.

Experiment 3 – Configured Device but NO Clock

Method: Configure the Spartan device with the ‘power_testing’ design and confirm operation as described on page 10 and the terminal display is that shown on page 11 (all functions disabled). Then set slide SW0 switch to ‘LOGIC 0’ such that LD0 is off and the INIT LED stops flashing. In this situation the 50MHz clock entering the design is being isolated such that the whole design is static (see page 17 for circuit diagram of clocks).

$$I_{CCINT} = 12.1\text{mA} \quad I_{CCAUX} = 11.5\text{mA}$$

($P_{INTERNAL} = 52\text{mW}$)

Experiment 4 – Suspend Mode

Method: Configure the Spartan device with the ‘power_testing’ design and confirm operation as described on page 10 and the terminal display is that shown on page 11. Then set SW4 switch (upper-right side of board) to the SUSPEND position such that the AWAKE is off and the INIT LED stops flashing. This has invoked the special power saving mode provided on the Spartan-3A device which effectively freezes the current state of the design so that once again it is “doing nothing” (see pages 37 and 38 for more details about suspend mode).

$$I_{CCINT} = 13.5\text{mA} \quad I_{CCAUX} = 1.4\text{mA}$$

($P_{INTERNAL} = 21\text{mW}$)

Power Theory – Part 2a

Having established the quiescent or static power dissipation of the device when configured with the 'power_testing' design it is now possible to move onto the measurement of dynamic power consumption for which the design has really been created. Although dynamic power consumption is based on solid theory, this is one of those areas where it is rather impractical to provide adequate data to predict results accurately. Unfortunately this leads many people to make guesses which are often wildly inaccurate which combined with other estimates often leads people to believe this whole subject is a bit of a 'dark art'! Hopefully this reference design will enable you to make real measurements and understand exactly what those measurements relate to such that your judgement of all things related to power dissipation will be improved. First we should review the theory of dynamic power consumption in digital circuits.

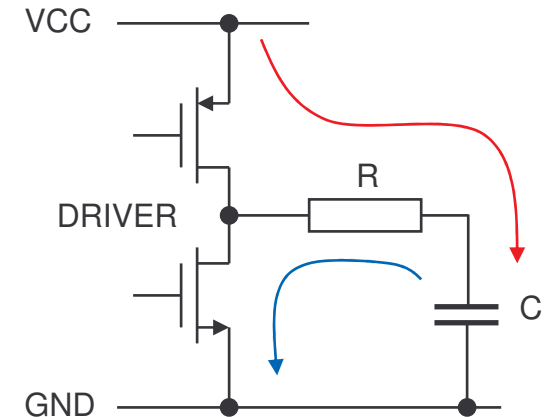
Dynamic Power

In an ideal world, only one transistor is turned on at a time in a CMOS logic switch.

During a **Low to High transition** the upper transistor is turned on and current flows from the supply to charge the capacitance of the load (C) and therefore this is when the supply has to deliver current. Due to resistance (R) the voltage rise is not instantaneous and results in what we refer to as 'switching delay'.

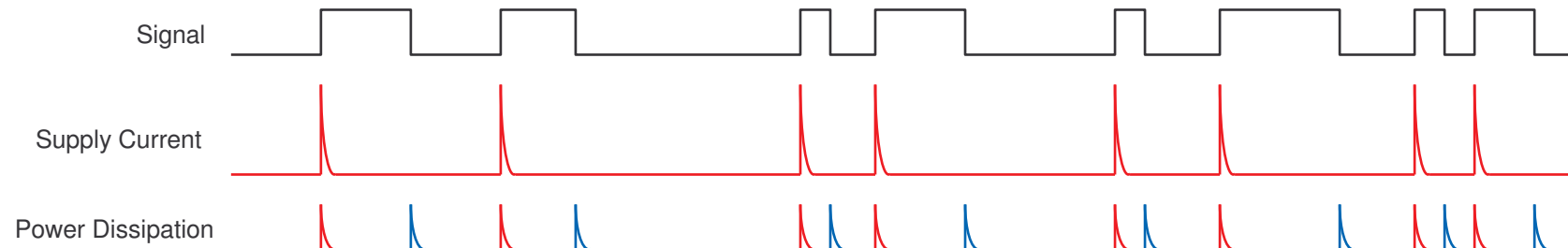
During a **High to Low transition** the lower transistor is turned on and the capacitor discharges with current flowing from the capacitor through 'R' to ground. Note that no current is drawn from the supply at this time.

The values of 'R' and 'C' in an FPGA design depend on the number and type of elements being connected as well as the interconnect use to join them. That is already a lot of variables making estimates less than easy.



The capacitive charging and discharging current, and hence power dissipation, is very 'spiky'. Indeed the current spikes can be very large albeit for a very short period of time and emphasizes the requirement for good decoupling capacitors. However it is not the peak current and power dissipation that are important to us but the average value of each. We therefore need to know the number of transitions that occur over a period of time to determine the average current or power. We typically use one second as the this averaging period and then talk about the frequency of the signal in Hertz (Hz).

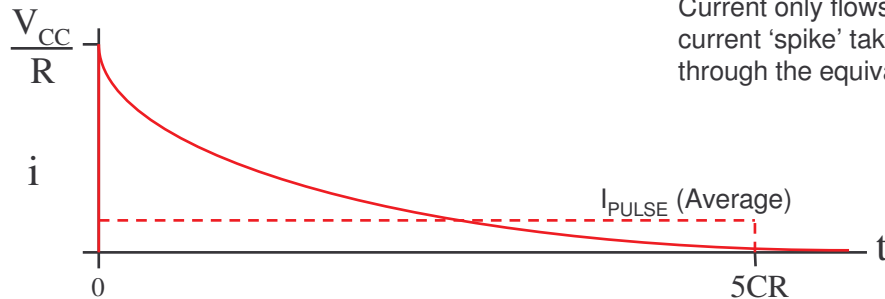
Hint – Note that as signal of 1MHz has 2 million transitions per second so be careful not to confuse frequency with transition rate or toggle rate.



Note that although current is only drawn from the supply during High to Low transitions, HALF of the power is dissipated in 'R' during the Low to High transition and HALF the power is dissipated in 'R' during the High to Low transition.

Power Theory – Part 2b

It is not immediately obvious that the dynamic power consumption is totally independent of the resistance 'R'. The resistance only effects the peak value of the current spikes and contributes to the switching delay. Power is only related to the value of the capacitance 'C', the supply voltage (squared) and the frequency of operation as shown below.



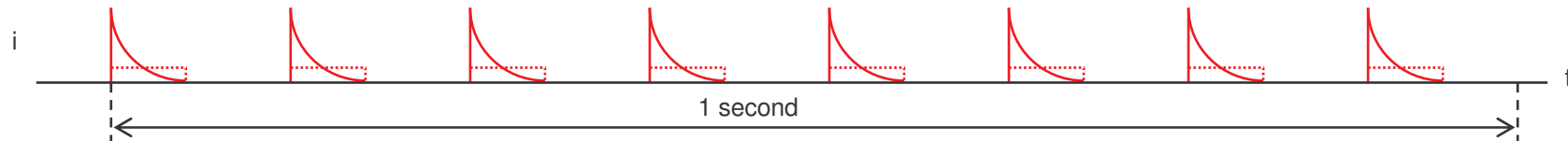
Current only flows from the power supply during Low to High transitions and in theory this current 'spike' takes an exponential form as the equivalent capacitance of the load charges through the equivalent resistance of the circuit. Hence the classic exponential equation.

Instantaneous current $i = \frac{V_{CC}}{R} e^{-\frac{t}{CR}}$

The average current which is flowing throughout the 'spike' is the most useful value and is derived by the integral of the exponential equation over five time constants (the time taken for the current to fall almost to zero).

$$I_{PULSE} = \frac{V_{CC}}{R} \int_0^{5CR} e^{-\frac{t}{CR}} \delta t = \frac{V_{CC}}{R} \left[\frac{e^{-\frac{t}{CR}}}{-\frac{1}{CR}} \right]_0^{5CR} = \frac{V_{CC} \cdot C \cdot R}{R} \left[-e^{-\frac{t}{CR}} \right]_0^{5CR} = V_{CC} \cdot C \left[1 - e^{-5} \right] = 0.9933 V_{CC} \cdot C$$

Removing the inaccuracy of the 5CR time constant we can actually see that the average current during a 'spike' is..... $I_{PULSE} = V_{CC} \times C$



The average current drawn from the supply is now defined by the number of current pulses over a longer period of time. So if we base that on pulses per second we can relate this directly to frequency.

$$I_{AVERAGE} = V_{CC} \times C \times f \quad \text{Hence power (VxI).....} \quad P_{AVERAGE} = V_{CC}^2 \times C \times f$$

Hint – Not all signals are nice regular clocks and this is a reason why we often consider the number of transitions per second rather than pure frequency. Just remember to divide signal transitions per second by 2 in order to obtain average frequency.

Measuring Dynamic Power

This is where the 'power_testing' design can help us to relate theory to reality. The theory tells us that power dissipation is related to voltage (squared) but for all practical purposes voltage is fixed. Which leaves us with capacitance and frequency. Capacitance depends on how the design logic is implemented, placed and routed into the Spartan-3A device which is certainly hard to predict (estimate) but can be analysed once the design is complete (i.e. XPower tool).

However, what no tool can really do easily is work out the frequency or transition rate of signals unless you tell it enough about how the logic will operate. Given the potentially high number of signals in a design this can be a very difficult and time consuming task unless you start making very general claims such as "all signals have a frequency of 20% of the clock" which is unrealistic on a good day and potentially very misleading on a bad day!

Frequency is Critical – The first way in which the 'power_testing' design helps to understand and measure power is that it allows you to clock the design at four different frequencies; 6.25MHz, 12.5MHz, 25MHz and 50MHz. On the next page you will see a detailed description of the clock generation and selection circuit which is part of the top level 'power_testing' design. Please begin your dynamic power measurements by determining the power consumption of this small by fast circuit.

Experiment 5 – Measure Power of Clocking Circuit

Method: First review the next slide to appreciate the circuit to be measured as well as review the clock options you have when conducting experiments. Configure the Spartan device with the 'power_testing' design and confirm operation as described on page 10. Then set slide SW1 switch to 'LOGIC 0' such that LD1 is off and the INIT LED stops flashing. In this situation the 50MHz clock is entering the clock generation and selection circuit but the selected clock is not being distributed to the rest of the design. Use slide switches SW2 and SW3 to select different frequencies.

With this set up, most of the design is static as it was in experiment 3 (page 13) and any increase in current is due to this small clock circuit. Although the circuit is small the 'signal' frequencies are all relatively high so the change in supply current is detectable.

Author's measurements

The measured figures in this table are the new base line for determining the dynamic current of other logic functions in future experiments.

	Total Measured	
	I _{CCINT} (mA)	I _{CCAUX} (mA)
6.25MHz	12.6	11.5
12.5MHz		
25MHz		
50MHz		

By subtracting the static measurements made in experiment 3 we can determine the dynamic current associated with the clock generation and selection circuit.

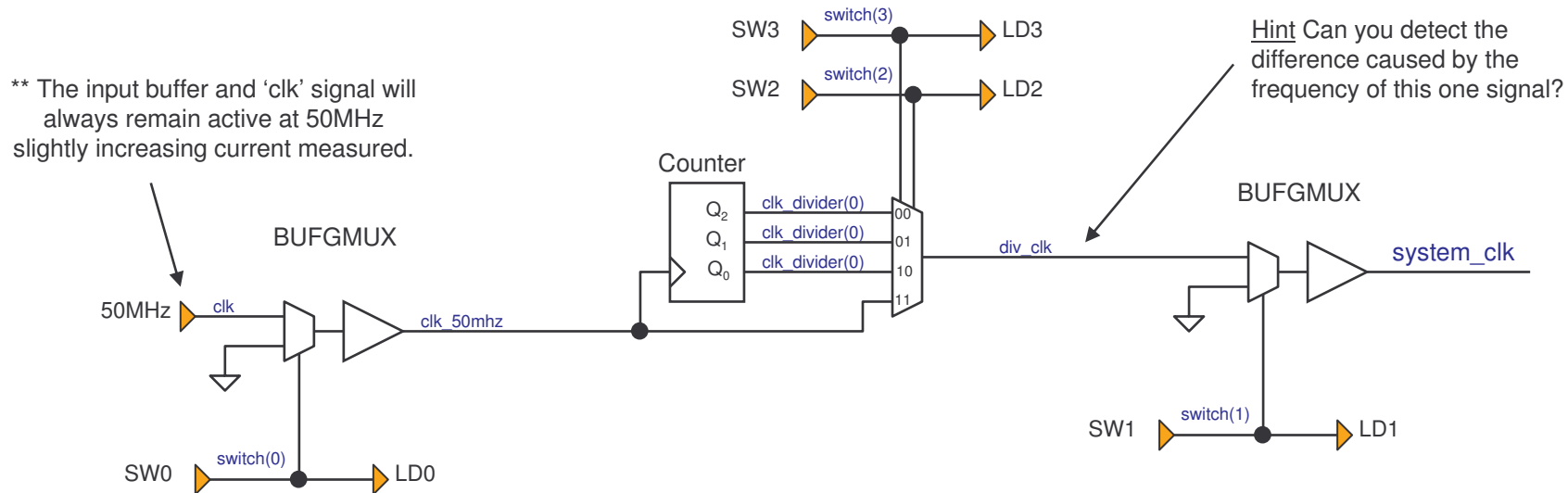
	Dynamic Current	
	I _{CCINT} (mA)	I _{CCAUX} (mA)
6.25MHz	0.5	0
12.5MHz		
25MHz		
50MHz		

Hint – This emphasizes that high frequency signals do impact current and power. When using power estimation tools it would be sensible to specify at least the high frequency signals with a high degree of accuracy.

Clock Control Circuit

Slide switch SW0 can be used to stop the 50MHz clock as it enters the device and allow the quiescent current of the Spartan-3A device to be measured. The clock supplied to the 'power modules' can be set to one of four frequencies; 6.25MHz, 12.5MHz, 25MHz or 50MHz using slide switches SW2 and SW3. This will enable you to evaluate the dynamic power consumption of various functions at different frequencies. However, it should be realised that this clock generation and control circuit will itself result in some power dissipation and therefore it is also possible to stop the selected clock from being distributed to the main system power modules using switch SW1 such that the supply current associated with this clock circuit can be determined (see previous page).

SW3	SW2	SW1	SW0	Effect
X	X	X	0	All clocks stopped – Quiescent current can be measured**.
X	X	0	1	System clock stopped – Current associated with this clock generation circuit can be determined.
0	0	1	1	System clock = 6.25MHz
0	1	1	1	System clock = 12.5MHz
1	0	1	1	System clock = 25MHz
1	1	1	1	System clock = 50MHz



Measuring PicoBlaze Power - 1

In this experiment we are going to measure the current and determine the dynamic power consumption of the PicoBlaze 8-bit micro-controller. We will be able to do this because the 'power_testing' design allows you to disable all other circuits (the default setting) and the previous experiments have already established the static power of the design in the Spartan-3A device as well as the incremental dynamic power required by the clock generation circuits.

On page 20 you can review how the PicoBlaze processor's input and output ports are connected to peripherals in each of the ten 'power_modules'. This may look a little complex but the important point to recognise is that all these peripherals will be static unless a command is entered at the PC terminal.

In addition to the 10 PicoBlaze processors, the small auxiliary circuit shown on page 21 will also be operating. Of course this will also contribute to the dynamic current but it is a small circuit in which most signals are of low frequency. More significantly there is only one of these auxiliary circuits compared with ten PicoBlaze processors. Later you will have enough knowledge to make a very accurate estimate of how much power the counters in this auxiliary circuit require but for now we will simply accept that our power figures for PicoBlaze will be slightly inflated.

Experiment 6 – Measure Power of Ten PicoBlaze Processors

Method: Configure the Spartan device with the 'power_testing' design and confirm operation as described on page 10.

It is important that the terminal display is that shown on page 11 showing that all other circuits are disabled.

Use slide switches SW2 and SW3 to select different frequencies and measure the current.

Author's measurements

This table shows the total current measured at the four different clock frequencies.

	Total Measured	
	I _{CCINT} (mA)	I _{CCAUX} (mA)
6.25MHz	17.9	11.5
12.5MHz	23.0	
25MHz	32.5	
50MHz	50.9	

By subtracting the total current measurements made in experiment 5 (page 16) we can determine the increase to dynamic current associated with 10 PicoBlaze Processors.

	Dynamic Current	
	I _{CCINT} (mA)	I _{CCAUX} (mA)
6.25MHz	5.1	0
12.5MHz	10.4	
25MHz	19.8	
50MHz	38.1	

Therefore the dynamic current of a single PicoBlaze processor is between 0.51mA and 3.81mA which translates to 0.6mW to 4.6mW.

Increase is non-linear – The dynamic current figures show a very strong correlation with the theoretical model. However you can see that the dynamic current is not quite doubling with each doubling of frequency. This is almost certainly a result of measurement induced error (see page 7)

Auxiliary Supply Remains Constant – It can be seen that the auxiliary supply current is not increasing in any noticeable way. Please confirm this observation for yourself, but to simplify this document I will now exclude auxiliary current measurements.

NOTE: The figures in the left hand table essentially become the base line when determining the dynamic current of the functions PicoBlaze will enable.

Measuring PicoBlaze Power - 2

We have just determined that the dynamic power consumption of a single PicoBlaze processor is 0.6mW to 4.6mW depending on clock frequency. A PicoBlaze processor consists of ~100 slices implementing a variety of familiar logic functions and 1 Block Memory storing the program and as such it is a nice representation of 'general logic'. The danger is that we may be too quick to assume that we have determined the typical power consumption for any 100 slices and a Block Memory and this is particularly tempting to do when we are not really sure what is happening at a low enough level. Remember that it doesn't matter how much logic is present but how that logic is operating; it is the average frequency of each signal which determines the power consumption.

```

Addr   Op-Code
10D    04000   read_from_UART: INPUT s0, status_in_port
10E    12004           TEST s0, rx_data_present
10F    3510D           JUMP Z, read_from_UART
    
```

This tiny abstract of the PicoBlaze program (pwr_ctrl.psm) shows what the processor is doing whilst it is waiting for a command. It simply checks to see if there is anything in the receiver UART buffer. The program address is hardly changing, only one register is used and only one ALU operation is executed.

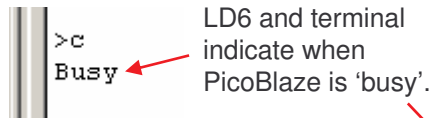
In fact there is more logic active inside PicoBlaze as part of its general operation but even so some features such as the scratch pad memory (formed of 16 slices) are definitely not being used and are therefore static. So what we will do now is make PicoBlaze work hard and measure the current again.

Experiment 7 – Measure Power of Ten PicoBlaze Processors Working Hard!

Method: Use exactly the same method as used in experiment 6 (previous page). Then enter the 'C' command and make your current measurements.

The 'C' Command – Calculation Test

This command invokes PicoBlaze to execute a routine that computes the 48-bit sum of all 24-bit numbers 000000-FFFFFF hex. This calculation is performed in a deliberately inefficient way (see 'pwr_ctrl.psm' for details) in order that the registers, ALU and scratch pad memory are all used. PicoBlaze will be 'busy' performing this calculation for ~15 seconds at 50MHz which translates to ~2 minutes at 6.25MHz and this should provide adequate time for you to make your current measurements.



LD7 indicates when PicoBlaze is 'ready'

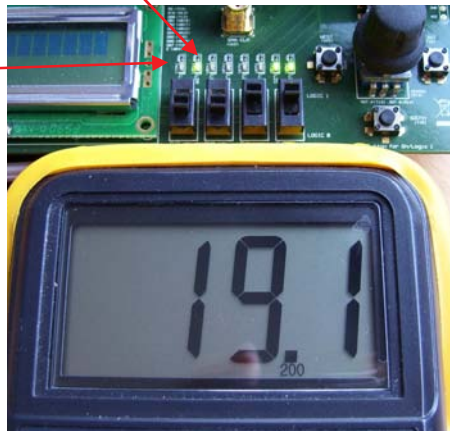


Photo shows SW2 and SW3 selecting 6.25MHz and the total measured current of $I_{CCINT} = 19.1\text{mA}$.

Author's measurements

10 PicoBlaze
~1000 Slices
and 10 BRAM


	Measured		Dynamic	
	I_{CCINT} (mA)		I_{CCINT} (mA)	
6.25MHz	19.1	17.9	6.5	5.1
12.5MHz	25.6	23.0	13.0	10.4
25MHz	36.2	32.5	23.4	19.8
50MHz	58.1	50.9	45.3	38.1

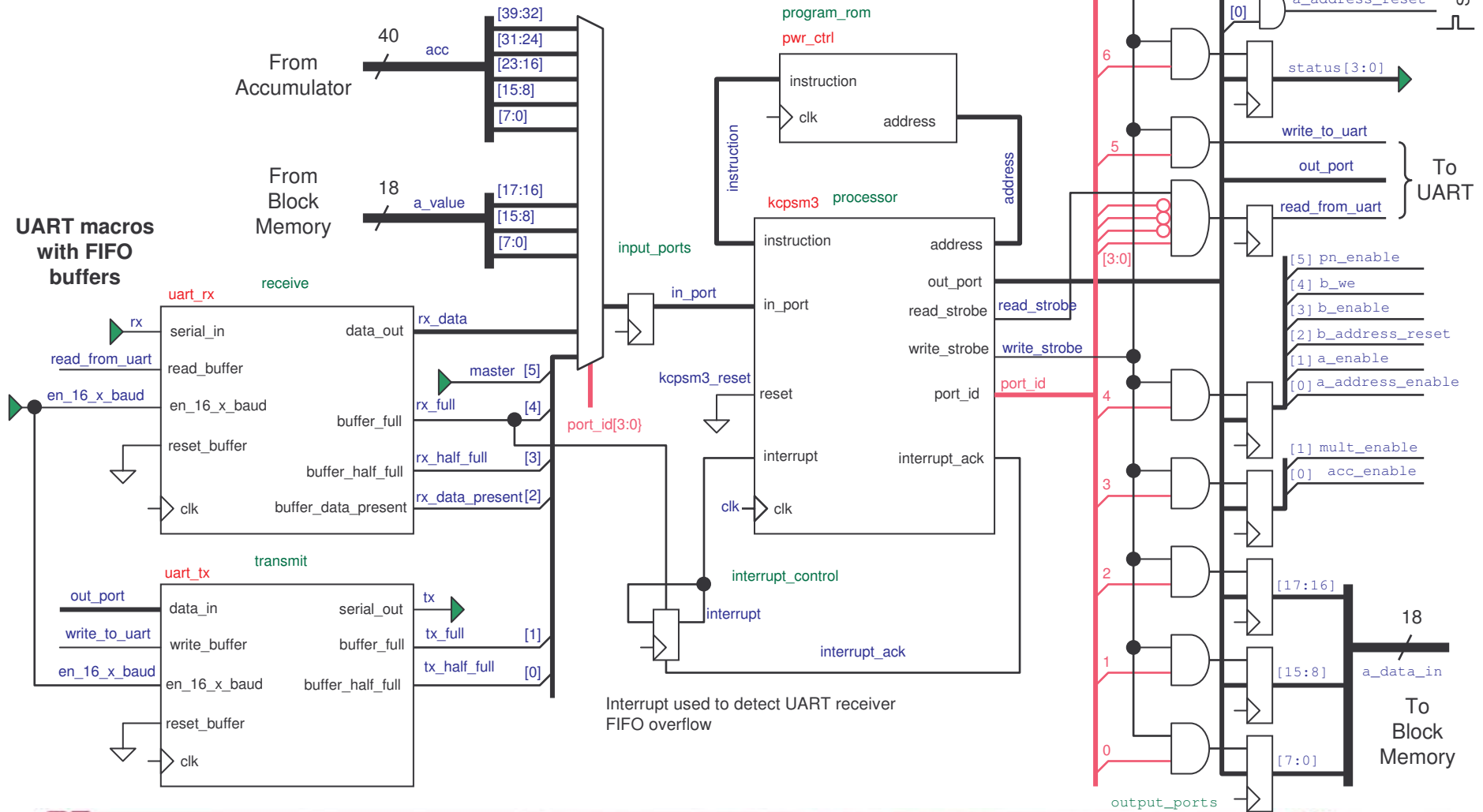
Blue figures for comparison show measurements from experiment 5 (PicoBlaze waiting for command)

Therefore the dynamic current of a single PicoBlaze 'working hard' is between 0.65mA at 6.25MHz and 4.53mA at 50MHz which translates to 0.8mW to 5.4mW.

When PicoBlaze is 'working hard' the supply current increases by up to 42% over the the idle state.

PicoBlaze Circuit Diagram

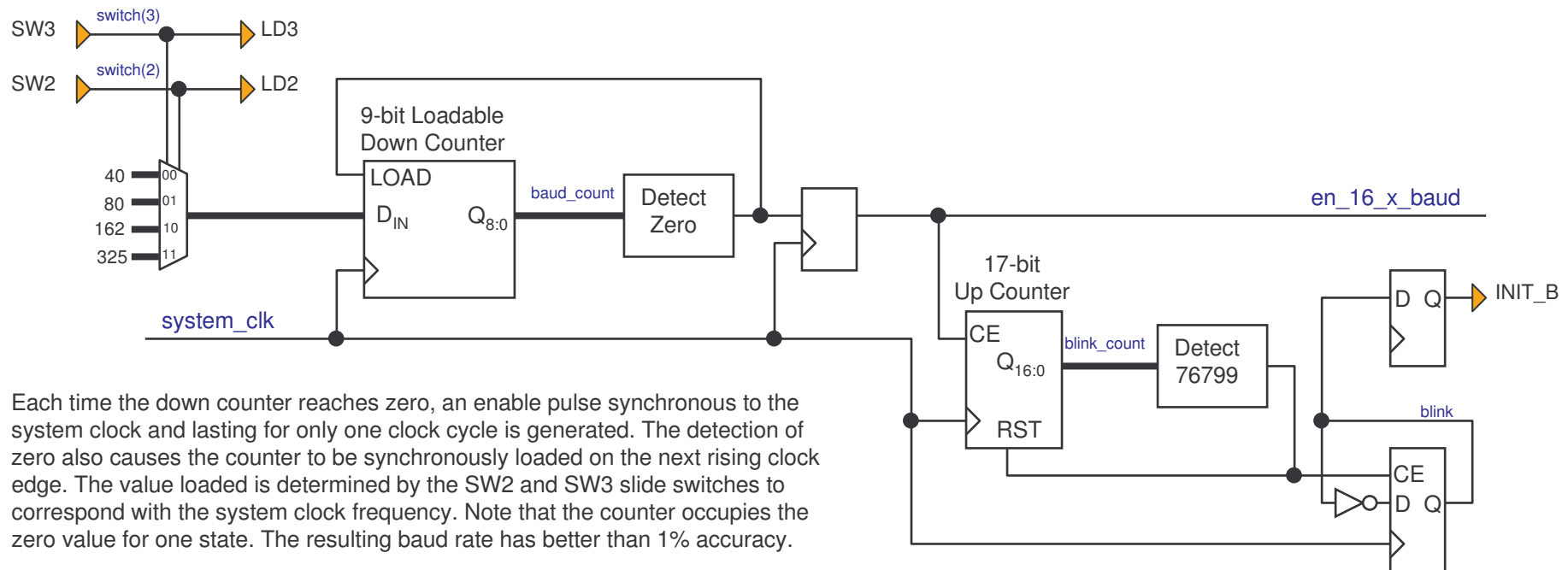
 **clk** All logic in the power module including PicoBlaze is clocked by the System Clock which will be 6.25MHz, 12.5MHz, 25MHz or 50MHz.
 Green arrows represent connections to and from the main system design.



Auxiliary Circuits

This page is for completeness of the reference design and not of general concern during our power measurements.

The PC with HyperTerminal and all the power modules are connected using a serial (UART) communication 'ring' operating at 9600 baud (see page 4). The UART macros contained in each power module require enable pulses at a rate which is 16 times higher than the desired baud rate (153,600Hz) and these are derived from the system clock. The slight complication is that the system clock frequency can be adjusted so the baud rate generator must also take this into account for the baud rate to remain a constant 9600. For this reason the baud rate generator is implemented only once and the enable signal distributed to all power modules. A further division of the baud enable pulses is used to blink the INIT LED to once per second regardless of the clock rate selected and confirm when the system clock is active.



Each time the down counter reaches zero, an enable pulse synchronous to the system clock and lasting for only one clock cycle is generated. The detection of zero also causes the counter to be synchronously loaded on the next rising clock edge. The value loaded is determined by the SW2 and SW3 slide switches to correspond with the system clock frequency. Note that the counter occupies the zero value for one state. The resulting baud rate has better than 1% accuracy.

SW3	SW2	System Clock Frequency	Count value	Counter States	en_16_x_baud pulse rate	Baud Rate
0	0	6.25MHz	40	41	152,439	9527
0	1	12.50MHz	80	81	154,321	9645
1	0	25.00MHz	162	163	153,374	9586
1	1	50.00MHz	325	326	153,374	9586

Simple counter divides the 'en_16_x_baud' by 76,800 and the toggle flip-flop divides by a further factor of two which is a total division of 153,600. This is used to blink the INIT LED approximately once per second.

Dynamic Power Measurements

You will now be able to use the 'power_testing' design to make accurate and meaningful measurements of various functions very rapidly. This document will lead you through some useful combinations and explain what the reference design has to offer but this is really where you can focus on the functions, frequencies and combinations that interest you the most. You are provided with all the source files as well as this document so you are encouraged to build on the methodology presented and measure the power of different circuits that you create yourself.

```
>p
OK

R - Reset
C - Calculation Test
F - Functional Test
1 - Fill BRAM 1,2,3...
2 - Fill BRAM +1,-1...
3 - Display BRAM Contents
4 - (0) A-Port BRAM Enable
5 - (0) A-Port Address Counter Enable
6 - (1) B-Port BRAM Enable
7 - (1) B-Port Address Counter Enable
8 - (1) B-Port BRAM Write Enable
P - (1) PN Generator Enable
M - (0) Multiplier Enable
A - (0) Accumulator Enable

>
```

The terminal display presents you with 14 options of which we have only used the 'C' command so far and 'R' simply resets the whole design (including memory contents).

Now we will use all the remaining commands which are used to control the logic functions and Block Memory introduced on page 4 and shown in full detail on the next page (page 23). Although this logic includes a multiplier and accumulator, please do not think that this design is only applicable to DSP designers because there are also counters and the Block Memory and any functions that remain disabled will not contribute to the current and can be ignored. Remember that you can always insert your own logic at a later stage.

As shown on page 4, there are 10 *identical* 'power_modules' connected in a ring. When you enter a command all the 'power_modules' will respond identically except for the small difference in start time resulting from the serial communication (this should not effect any current measurements). The last module in the ring is identified as the 'master' which means that it is the one to drive the terminal display.

LED 'LD7' is the logical AND of the 'ready' status from all 10 modules and therefore is only illuminated when all modules are 'ready'. 'LD6' is the logical OR of the 'busy' status from all 10 modules and therefore is illuminated when any of the modules are still 'busy' (see page 19).

The concept of the design is that we will use the terminal commands to enable and disable particular features in various combinations and measure the *change* in the supply current. It is therefore very useful to have your own figures from experiment 5 to hand as these figures cover all of the device and parts of the design that we are using to control the next features. However, it is so easy to toggle features on and off that you will be able to determine the associated increase and decrease in current almost immediately.

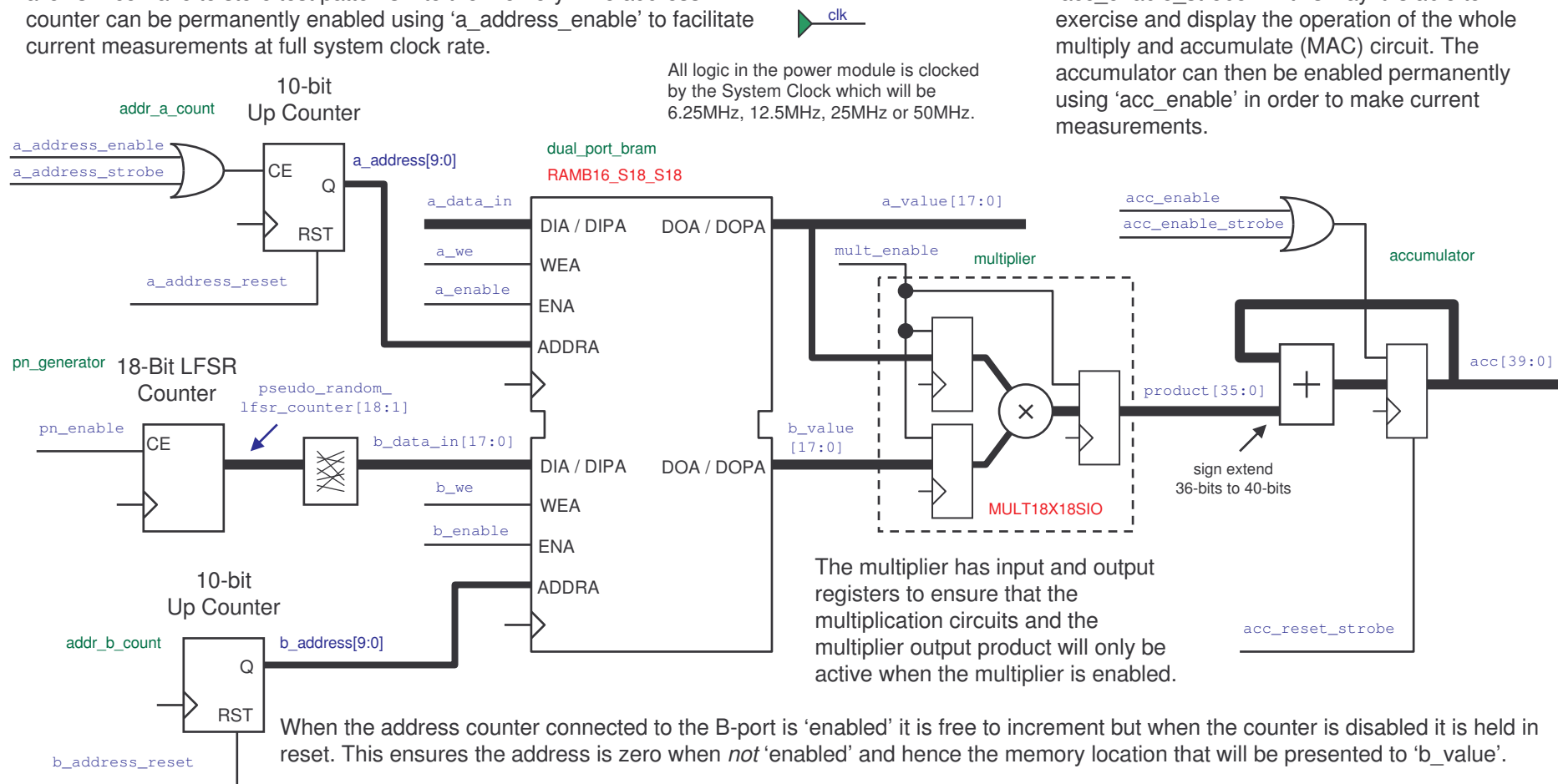
Hint – Over a long period of time the ambient temperature may change and this will cause a small variations to the total current figures measured. Although small, they may be larger than the changes resulting from the functions being enabled or disabled so the more immediate comparison is beneficial.

Remember, all functions are replicated 10 times so the difference in current measured is magnified and you must divide by 10 when calculating the dynamic current of power of individual instances of functions. The following pages illustrate this several times.

BRAM & MAC Circuit Diagram

PicoBlaze can reset the A-port address counter and then increment it using 'a_address_strobe'. It can then read the 'a_value' data from the memory and display it or it can write a new 18-bit value 'a_data_in'. This allows PicoBlaze to store test patterns into the memory. The address counter can be permanently enabled using 'a_address_enable' to facilitate current measurements at full system clock rate.

PicoBlaze can reset the 40-bit accumulator to ensure a known starting point and then enable the accumulator for single clock cycles using 'acc_enable_strobe'. In this way it is able to exercise and display the operation of the whole multiply and accumulate (MAC) circuit. The accumulator can then be enabled permanently using 'acc_enable' in order to make current measurements.



All logic in the power module is clocked by the System Clock which will be 6.25MHz, 12.5MHz, 25MHz or 50MHz.

The multiplier has input and output registers to ensure that the multiplication circuits and the multiplier output product will only be active when the multiplier is enabled.

When the address counter connected to the B-port is 'enabled' it is free to increment but when the counter is disabled it is held in reset. This ensures the address is zero when *not* 'enabled' and hence the memory location that will be presented to 'b_value'.

The linear feedback shift register (LFSR) counter can be individually enabled or disabled to allow current measurements to be made. The 18-bit output of the LFSR counter is applied to the B-port data of the memory (via a bit scrambled connection) in order to provide pseudo random data values. This can be used to measure current during memory writing as well as exercise the MAC.

Functional Tests

Before you can trust any measurements it is useful to see that the logic which is to be analysed really does operate in the way you would expect. Indeed it is absolutely vital that the signal activity is known as this determines the 'average frequency' which is so critical to dynamic power consumption. So some of the commands are simply to verify the functionality of the logic circuit you will be measuring.

Command '3' - Display BRAM Contents

The Block Memory (BRAM) is configured as 1024 locations of 18-bits. The 'Display BRAM Contents' command will read and display all 1024 values as 5 digit hexadecimal values starting at location zero and ending at location 1023 (3FF hex). The display is 128 lines of 8 values so read quickly or use the scroll buffer to review it! Following device configuration or the 'R' command the memory will be completely zero as shown. Zero data is useful in some ways but not for proving functionality!

```
>3
00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000
...
```

Command '1' - Fill BRAM 1,2,3...

On receiving this command PicoBlaze write the ascending values 1, 2, 3, 4, 5 etc into the memory starting at address location zero. This is very simple and predictable test pattern and one that is easy to work with. After this command has been used you can confirm that the memory has been programmed using the 'Display BRAM Contents' command as shown below.

```
>3
00001 00002 00003 00004 00005 00006 00007 00008
00009 0000A 0000B 0000C 0000D 0000E 0000F 00010
00011 00012 00013 00014 00015 00016 00017 00018
.
.
.
003F1 003F2 003F3 003F4 003F5 003F6 003F7 003F8
003F9 003FA 003FB 003FC 003FD 003FE 003FF 00400
```

Command 'F' - Functional Test

This command causes PicoBlaze to enable the logic functions and memory for 20 'steps' such that values can be read and displayed to confirm operation.

The address counter to the B-port is disabled and held reset which means that the value stored at memory location zero is always being read and applied to the 'B' input of the multiplier. If the '1,2,3' pattern is loaded then this is conveniently the value '1' which makes multiplication easy!

The address counter to the A-port is initially reset and then incremented for each 'step'. Therefore the 'A' input to the multiplier is provided with different values which are displayed under 'BRAM' as shown here.

The multiplier computes $A \times B$ and is applied to the accumulator which is enabled for one 'step' (one clock cycle) and the 40-bit accumulated value (10-digit hex) is displayed under 'Accumulator' on the right.

E.g. $(1 \times 1) + (1 \times 2) + (1 \times 3) + (1 \times 4) = 10$ (0A hex)
as shown on the 4th line of the display here.

```
>f
BRAM  Accumulator
00001 000000001
00002 000000003
00003 000000006
00004 00000000A
00005 00000000F
00006 000000015
00007 00000001C
00008 000000024
00009 00000002D
0000A 000000037
0000B 000000042
0000C 00000004E
0000D 00000005B
0000E 000000069
0000F 000000078
00010 000000088
00011 000000099
00012 0000000AB
00013 0000000BE
00014 0000000D2
```

Dynamic Power of Binary Counters

The 'power_module' contains two 10-bit binary counters which are used to address the 'A' and 'B' ports of the Block Memory. In this experiment you will leave all other functions disabled so that they remain static and only enable the counters.

```
>5
OK

R - Reset
C - Calculation Test
F - Functional Test
1 - Fill BRAM 1,2,3...
2 - Fill BRAM +1,-1...
3 - Display BRAM Contents
4 - (0) A-Port BRAM Enable
5 - (1) A-Port Address Counter Enable
6 - (0) B-Port BRAM Enable
7 - (0) B-Port Address Counter Enable
8 - (0) B-Port BRAM Write Enable
P - (0) PN Generator Enable
M - (0) Multiplier Enable
A - (0) Accumulator Enable

>
```

Command '5' – A-Port Address Counter Enable

Simply enter the command and the menu/status will confirm that the 10-bit binary counter addressing the A-port of the memory is enabled. Enter '5' again and it will be disabled. By observing your ammeter reading and using the command to enable and disable the counter you can quickly determine the current increase cause by the additional dynamic activity.

Author's measurements

This table shows the difference in measured current noted as the '5' command was used to toggle the **10** 'A-port Address' counters between enabled and disabled states.

	Dynamic I_{CCINT} (mA)
6.25MHz	0.1
12.5MHz	0.3
25MHz	0.5
50MHz	1.0

Command '7' – B-Port Address Counter Enable

This enables and disables the 10-bit counter addressing the B-port of the memory.

10 'B-port Address' counters

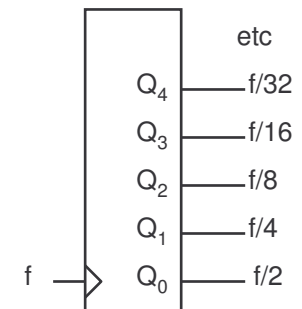
	Dynamic I_{CCINT} (mA)
6.25MHz	0.2
12.5MHz	0.3
25MHz	0.6
50MHz	0.9

Using commands '5' and '7' to simultaneously enable all **20 counters** binary counters. The result is reassuringly close to being the sum of the two separate tables ☺

	Dynamic I_{CCINT} (mA)
6.25MHz	0.3
12.5MHz	0.6
25MHz	1.1
50MHz	2.1

~0.1mA per counter at 50MHz

Hint – A free running binary counter has very predictable signal toggle rates (transitions or frequencies). The least significant bits are responsible for the majority of the dynamic power so the size of a counter is almost irrelevant.



Dynamic Power of LFSR Counters

The 'power_module' contains one 18-bit binary linear feedback shift register (LFSR) counter which can be used to generate pseudo random data that can then be written into the B-port of the Block Memory. In this experiment we will only look at the LFSR counter and will leave all other functions disabled.

```
>p
OK

R - Reset
C - Calculation Test
F - Functional Test
1 - Fill BRAM 1,2,3...
2 - Fill BRAM +1,-1...
3 - Display BRAM Contents
4 - (0) A-Port BRAM Enable
5 - (0) A-Port Address Counter Enable
6 - (0) B-Port BRAM Enable
7 - (0) B-Port Address Counter Enable
8 - (0) B-Port BRAM Write Enable
P - (1) PN Generator Enable
M - (0) Multiplier Enable
A - (0) Accumulator Enable

>
```

Command 'P' – PN Generator Enable

Simply enter the command and the menu/status will confirm that the 18-bit LFSR counter Used to generate pseudo random data is enabled. Enter 'P' again and it will be disabled.

Author's measurements

This table shows the difference in measured current noted as the 'P' command was used to toggle the 10 'PN Generator' LFSR counters between enabled and disabled states.

	Dynamic I_{CCINT} (mA)
6.25MHz	0.5
12.5MHz	1.1
25MHz	2.0
50MHz	3.5

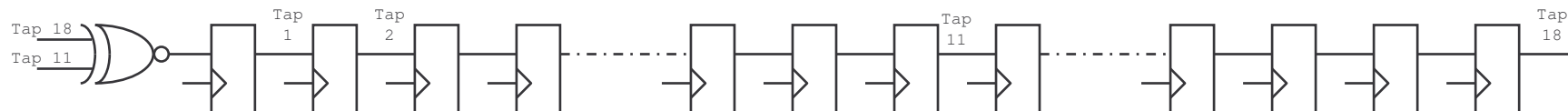
~0.35mA per counter at 50MHz

It can be seen that the dynamic current associated with the 18-bit LFSR counter is approximately 3.5 times greater than that of the 10-bit binary counters.

We might like to attribute some of the difference to the larger number of bits, but as discussed on the previous page, the addition of more bits to the binary counter would increase the current by less than 1/1024 because the frequency of the most significant bits would be so low.

We might consider that the equivalent capacitance of the connections and loading on the LFSR counter is higher. Well admittedly there would be some variation but it is most unlikely to be this significant given the direct connection to the Block Memory.

Which means that this increased current (and power consumption) must be entirely attributed to the frequency of the signals. This is actually very reasonable considering that an LFSR counter is shift register. It means that over time all outputs have the same number of transitions (same frequency) and so any increase in the size of an LFSR counter will proportionally increase the dynamic current and power which is total different to the binary case.



Block Memory (BRAM) Power

The 'power_module' contains one 18K-bit Block Memory (BRAM) configured as 1204×18-bits dual port. The memory has controls on each port. The 'ENA' and 'ENB' inputs are the global enable controls for each port. It is only possible to perform any operation, including reads, when the enable signal for a port is active (High). The 'WEA' and 'WEB' are the write enable inputs for each port. When the write enable is active (High) at the same time as the global enable then data presented to data input will be stored at the specified address.

There are several commands associated with controlling the memory. So we will review each of them and consider how they can be used in combinations.

Command '4' – A-Port BRAM Enable

and

Command '6' – B-Port BRAM Enable

These commands can be used to independently enable and disable each port of the memory

Experiment – With all other functions disabled, enable each port or the BRAM and note any change in current. Note that all signals to and from the BRAM except the clock will be static so this is a measurement of the memory itself.

Dynamic difference observed for 10 BRAMs

Author's measurements at 50MHz only

A-Port BRAM Enable $I_{CCINT} = 10.6\text{mA}$

B-Port BRAM Enable $I_{CCINT} = 10.2\text{mA}$

A-Port and B-Port BRAM Enable $I_{CCINT} = 20.3\text{mA}$

} Approximately 1mA per BRAM port

Hint – If low power consumption or current drain is important in your applications then seriously consider only enabling the BRAM on the clock cycles you actually need to read or write it. Many applications only read or write memory sporadically or in bursts so try to include the 'EN' signal in the control logic.

BRAM Power- Reading - 1

We can read the contents from the Block Memory (BRAM) using various combinations of the commands.

**Author's measurements at
50MHz only**

Method

Select the frequency of interest using SW2 and SW3 and ensure the system is reset by using the 'R' command.
Note the supply current with all additional circuits disabled.

$I_{CCINT} = 50.8\text{mA}$

Use command '5' and note the increase in supply current due to the 10 A-port address counters running (see page 25).
Then disable the counter again using command '5'.

$I_{CCINT} = 51.8\text{mA}$

Use command '4' and observe the increase in supply as the A-port on 10 BRAMs are enabled (see previous page).

$I_{CCINT} = 61.5\text{mA}$

Then, leaving the BRAM enabled, use command '5' to once again enable the 10 A-port address counters which will this time repeatedly scan read the memory contents out of the memory.

$I_{CCINT} = 62.7\text{mA}$

From these four current measurements we can determine the following:-

10 Address counters increase current by $52.0 - 51.8 = 1.0\text{mA}$ (as on page 25)

10 BRAM ports increase current by $61.5 - 50.8 = 10.7\text{mA}$ (as on previous page)

Scan reading 10 BRAM ports increases current by $62.7 - 50.8 - 1.0 - 10.7 = 0\text{mA}$ i.e. Reading BRAM dissipates no more power!

Can this be right?

The increase in dynamic current when the BRAM was enabled by command '4' is the same as that measured on the previous page when the BRAM was enabled without the address counter causing the memory contents to be scanned. The reason for this is that the BRAM is completely empty (all values are zero) so all the data outputs of the BRAM are still static and there is no real addition to dynamic activity.

Command '2' – Fill BRAM +1,-1...

This command programs the BRAM in each 'power_module' with the alternating value of +1 and -1 (twos complement). Once programmed in this way the act of scan reading the memory will result in 17 of the 18-bits alternating between High and Low states which is a signal frequency of half the system clock rate which is the 'worst' data pattern you could have when it comes to generating power!

```
>3
00001 3FFFF 00001 3FFFF 00001 3FFFF 00001 3FFFF
00001 3FFFF 00001 3FFFF 00001 3FFFF 00001 3FFFF
00001 3FFFF 00001 3FFFF 00001 3FFFF 00001 3FFFF
00001 3FFFF 00001 3FFFF 00001 3FFFF 00001 3FFFF
```

Command '3' displays the memory contents.

+1 = 00 0000 0000 0000 0001

-1 = 11 1111 1111 1111 1111

Note: 17-bits change
between adjacent locations
when being scanned.

BRAM Power- Reading - 2

So now repeat the measurements with the BRAM programmed with this extreme data pattern.

Hint – Command 'R' will also reset the memory contents to all zero so remember to use command '3' to confirm what data values you are working with.

Following Command '2' to Fill BRAM with +1,-1... pattern

Author's measurements at 50MHz only

All additional logic disabled $I_{CCINT} = 51.2\text{mA}$

'5' – **A-port** Address counters enabled $I_{CCINT} = 52.1\text{mA}$

'4' – **A-port** of BRAMs enabled $I_{CCINT} = 61.7\text{mA}$

'4' + '5' – **A-port** of BRAM being scan read $I_{CCINT} = 65.9\text{mA}$

Note – Small fluctuations in total current values occur over time (temperature) so quick local comparisons ensure most accurate measurements

10 Address counters increase current by $52.1 - 51.2 = 0.9\text{mA}$

10 BRAM ports increase current by $61.7 - 51.2 = 10.5\text{mA}$

Scan reading alternating High-Low data pattern 10 BRAM ports increases current by $65.9 - 51.2 - 0.9 - 10.5 = 3.3\text{mA}$

You can see a very slight increase of 0.7mA when the 10 BRAM's are enabled whilst holding this extreme data pattern. But the main difference we can observe is when the ± 1 data pattern is being read. Note that is 17 outputs on each of 10 BRAM's all with a frequency of 25MHz (half the system clock rate). This illustrates why you have to consider the data values being read and how they determine the frequency of the output signals.

So is the 'B-port' the same? Well just repeat the experiment using the B-port commands.....

Author's measurements at 50MHz only

All additional logic disabled $I_{CCINT} = 51.2\text{mA}$

'7' – **B-port** Address counters enabled $I_{CCINT} = 52.3\text{mA}$

'6' – **B-port** of BRAMs enabled $I_{CCINT} = 61.0\text{mA}$

'6' + '7' – **B-port** of BRAM being scan read $I_{CCINT} = 63.0\text{mA}$

10 Address counters increase current by $52.3 - 51.2 = 1.1\text{mA}$

10 BRAM ports increase current by $61.0 - 51.2 = 9.8\text{mA}$

Scan reading alternating High-Low data pattern 10 BRAM ports increases current by $63.0 - 51.2 - 1.1 - 9.8 = 0.9\text{mA}$

Although we can see some slight variations in the figures for the address counters and the BRAM port enable these are not surprising given the resolution of the meter used by the author and the variations to be expected with temperature over time. However it is very clear that there is a larger increase in dynamic power associated with reading the A-port than the B-port. If you look back at the schematics on pages 20 and 23 you will see that PicoBlaze can also read the BRAM contents and it does this using the A-port. Hence the A-port has a greater effective capacitive load resulting in greater power dissipation. This further emphasizes how challenging it is to make accurate power estimates at an early stage of a design.

BRAM Power - Reading - 3

In this final experiment you can confirm the power associated with reading a BRAM using pseudo random data. On the following page we will investigate the power dissipated during writing of BRAM which will use the same mechanism but for now just follow the method shown or conduct your own experiments and measurements based on a similar scheme.

Method

Select the frequency of interest using SW2 and SW3 and ensure the system is reset using the 'R' command.

Use commands '6', '7', '8' and 'P' to simultaneously enable the BRAM B-port, B-port Write, B-Port Address Counter and the PN Generator (see terminal picture on next page). With all these features enabled the memory is written with random data. Then repeat commands '6', '7', '8' and 'P' to disable all the features again (do not use 'R' as this will clear the memory).

Use command '3' to observe the BRAM contents which should look something like this. All 10 BRAMs will contain slightly different pseudo random data due to the time delay reacting to your commands and this only enhances the average current measured.

```
>3
34781 007ED 0B62F 1FE24 1D974 2D95A 26B9A 3219D
3A6C9 12EE7 19477 2DE62 05BB6 2D31D 2ED8C 1ABB9
33E5D 3E4F6 399E3 05EDB 26737 395AC 0F8E9 17D13
34778 233EE 1F28F 1EF45 18F7E 2B87E 3FA92 34BD5
```

For a best average, I chose to scan read both the 'A' and 'B' ports at the same time.....

All additional logic disabled	$I_{CCINT} = 51.2\text{mA}$
'5' + '7' – Both Address counters enabled	$I_{CCINT} = 53.1\text{mA}$
'4' + '6' – Both ports of BRAMs enabled	$I_{CCINT} = 70.3\text{mA}$
'4' + '5' + '6' + '7' – Scan read of both BRAM ports	$I_{CCINT} = 75.2\text{mA}$

Author's measurements at 50MHz only

20 Address counters increase current by $53.1 - 51.2 = 1.9\text{mA}$

20 BRAM ports increase current by $70.3 - 51.2 = 19.1\text{mA}$

Scan reading random data from **20** BRAM ports increases current by $75.2 - 51.2 - 1.9 - 19.1 = 3\text{mA}$

~0.2mA per counter

~1mA per BRAM port (enable)

~ 0.15mA per BRAM port (read)

We can observe that pseudo random data results in something closer to half the signal frequency of the extreme ± 1 pattern or close to one quarter of the system clock frequency.

Hint - Enabling a BRAM port (EN=1) contributes more to current and power than the actual act of reading the data values so it really is worth controlling the enable to the BRAM such that it is only consuming power when reads are actually being performed.

BRAM Power- Writing

The '1' and '2' commands instruct PicoBlaze to store two simple data patterns into the Block Memory (BRAM) and the 'R' command can be used to clear the memory. Although these commands do involve writing to the BRAM, the operations take place over such a short period of time and using such a low write frequency that any measurements of supply current are difficult and virtually impossible to associate with the specific act of writing. For this reason the data input to the B-port of the BRAM is connected to an LFSR counter which can be used to generate pseudo random data values suitable for writing (see circuit diagram on page 23). You can write data values to the same address or enable the 'B' address counter to write to successive locations.

Method

Select the frequency of interest using SW2 and SW3.

Use commands '6', '7' and 'P' to enable the B-port of the memory, the associated address counter and the PN Generator connected to the data input. Note the current required by all the logic operating so far.

Use command '8' and observe the increase in supply as the B-port write enable is activated. (You may wish to enable and disable several times to take average readings as there will be some fluctuation resulting from the random data).

The terminal display on the right shows when all the B-port features are enabled.

```
R - Reset
C - Calculation Test
F - Functional Test
1 - Fill BRAM 1,2,3...
2 - Fill BRAM +1,-1...
3 - Display BRAM Contents
4 - (0) A-Port BRAM Enable
5 - (0) A-Port Address Counter Enable
6 - (1) B-Port BRAM Enable
7 - (1) B-Port Address Counter Enable
8 - (1) B-Port BRAM Write Enable
P - (1) PN Generator Enable
M - (0) Multiplier Enable
A - (0) Accumulator Enable
```

Author's measurements at 50MHz only

'6' + '7' + 'P' – Scan reading B-port plus PN Generator running (×10) $I_{CCINT} = 66.2\text{mA}$

'6' + '7' + 'P' + '8' – Writing random data into B-port (×10) $I_{CCINT} = 67.3\text{mA}$ ~ 0.1mA per BRAM port (write)

We can see that the increase in supply current when writing is 1.1mA which is ~0.1mA per BRAM port. Again this is a small increase compared with the additional 1mA consumed when enabling the port of the BRAM.

Hint - Enabling a BRAM port (EN=1) contributes more to current and power than the actual act of writing data values so it really is worth controlling the enable to the BRAM such that no power is consumed when not actually writing or reading data. Note that if a port is used only to write data (e.g. the write port of a FIFO), then it would be better to tie the write enable (WE) control High and use the enable (EN) control to write data.

Multiplier Power - 1

In each 'power_module' a multiplier is connected to the outputs of the Block memory (see page 23). This allows the multiplier to be operated whilst having various data values applied to its inputs. The dynamic power dissipation observed when the multiplier is enabled will be a combination of two factors. The first is the internal operation of the multiplier as it computes the product of the two inputs and the second will be the driving of the outputs as the product is routed to the accumulator. In this design it is not possible to separate these two factors but you may wish to alter the design such that the output register of the multiplier can be separately enabled and disabled.

Multiplier Experiment 1 – Zero Data

Select the frequency of interest using SW2 and SW3 and use the 'R' command to reset all logic and clear the memory.

Use commands '4', '5', '6' and '7' to enable both ports of the BRAM and their associated address counters such that the memory contents are being scan read and applied to the multiplier input ports. Note the supply current.

Use the 'M' command to enable and disable the multiplier and observe any increase in supply current.

Since all values are zero then it is not surprising that the increase is very small. The increase of ~50µm per multiplier is probably associated with the input and output registers (72 flip-flops) responding to the clock input even though the values remain constant.

Multiplier Experiment 2 - ±1 Pattern on A-port with B-port=+1

Select the frequency of interest using SW2 and SW3 and use the 'R' command to reset all logic and clear the memory.

Use command '2' to fill the memory with a +1, -1 data pattern.

Use commands '4', '5' and '6' to enable both ports of the BRAM but only enable the address counter connected to the A-port. In this way the data applied to the A-port is the alternating ±1 values but the input to the B-port remains at +1.

Use the 'M' command to enable and disable the multiplier and observe the increase in supply current as the product also alternates between +1 and -1. Note that the product is 36-bits and the ±1 values mean that 35-bits are toggling between Low and High states at half the system clock frequency.

Multiplier Experiment 2 - ±1 Pattern on B-port with A-port=+1

As above but replacing command '5' with command '6'.

**Author's measurements at
50MHz only**

$I_{CCINT} = 73.1\text{mA}$

$I_{CCINT} = 73.6\text{mA}$ (+0.5mA)

$I_{CCINT} = 74.9\text{mA}$

$I_{CCINT} = 85.2\text{mA}$ (+10.3mA)
~1mA per Multiplier

$I_{CCINT} = 72.4\text{mA}$

$I_{CCINT} = 83.2\text{mA}$ (+10.8mA)
~1.1mA per Multiplier

Multiplier Power - 2

Multiplier Experiment 4 – Random Data

Select the frequency of interest using SW2 and SW3.

Use commands '6', '7', '8' and 'P' to simultaneously enable the BRAM B-port, B-port Write, B-Port Address Counter and the PN Generator (see page 31). With all these features enabled the memory is written with random data. Then repeat commands '6', '7', '8' and 'P' to disable all the features again (do not use 'R' as this will clear the memory).

(Command '3' allows you to confirm the contents of the memory)

Use commands '4', '5', '6' and '7' to enable both ports of the BRAM and enable both the address counters such that random data is being read from both memory ports and applied to the multiplier. Note the supply current.

$$I_{CCINT} = 75.2\text{mA}$$

Use the 'M' command to enable and disable the multiplier and observe the increase in supply current as the product of the random numbers is computed and the product routed to the accumulator..

$$I_{CCINT} = 87.0\text{mA} \quad (+11.8\text{mA})$$

~1.2mA per Multiplier

In this case the frequency of the multiplier output signals (the product) will be lower and most likely close to quarter of the system clock rate. However the actual multiplication process is working very hard.

The product arriving at the accumulator may be causing some additional power to be dissipated in the accumulator. However the accumulator is reset by the 'R' command and therefore all additions being performed in the accumulator logic are zero + product which result in no carry operations and the accumulator is otherwise disabled. Hence this dissipation is almost certainly very low indeed. You may wish to prove this by inserting an additional register between the multiplier and the accumulator which you can disable when performing the multiplier current measurements.

**Author's measurements at
50MHz only**

Accumulator Power - 1

A 40-bit accumulator is connected to the output of the multiplier in each 'power_module' (see page 23) it should not be surprising to realise that we use all the other circuits to feed the accumulator with various interesting data test patterns. The output of the accumulator can be read by PicoBlaze (see page 24) but this read can not be performed at the full clock rate required when making supply current measurements. However, the connection of the accumulator to the PicoBlaze input port still presents a load on the accumulator outputs and means that the measurements taken are representative of a typical circuit.

Accumulator Experiment 1 - ± 1 Pattern

Select the frequency of interest using SW2 and SW3 and use the 'R' command to reset all logic and clear the memory.

Use command '2' to fill the memory with a +1, -1 data pattern.

Use the 'F' command to see how the accumulator will respond. Although both ports of the BRAM are enabled, the address to the B-port remains disabled and location zero of the memory containing +1 is applied to the multiplier. The address to A-port is incrementing and this applies the +1, -1 alternating sequence to the other input of the multiplier. This in turn means that the output of the multiplier is the alternating +1, -1 sequence which causes the accumulator to just increment and decrement as shown.

```
>f
BRAM Accumulator
00001 0000000001
3FFFF 0000000000
00001 0000000001
3FFFF 0000000000
00001 0000000001
3FFFF 0000000000
```

Author's measurements at 50MHz only

$$I_{CCINT} = 85.1\text{mA}$$

$$I_{CCINT} = 94.2\text{mA} \quad (+9.1\text{mA})$$

$$\sim 0.9\text{mA per Accumulator}$$

Now use the commands '4', '5', '6' and 'M' to enable the multiplier, both BRAM ports and the A-port address counter in the same way that the 'F' command did but this time at the system clock rate. Note the supply current.

Use the 'A' command to enable the accumulator and observe the increase in supply current.

In this experiment we know that most outputs from the accumulator are static so the current increase is almost entirely attributable to the operation of the accumulator and carry chain itself. The ± 1 is the most demanding exercise possible for the carry chain and therefore this represents the worst case for the accumulator logic itself.

Hint - The 40-bit accumulator is implemented using 20 logic slices of the Spartan-3A device and this would be the same in Spartan-3E devices. If you have a DSP intensive design which would benefit from many accumulators operating at higher frequencies then the Spartan-3ADSP devices have dedicated 48-bit accumulator logic as part of the DSP blocks (which also contain the multipliers). As dedicated functions these elements will be faster offer the potential for lower power dissipation.

Accumulator Power - 2

Accumulator Experiment 2 – Random Data

Select the frequency of interest using SW2 and SW3.

Use commands '6', '7', '8' and 'P' to simultaneously enable the BRAM B-port, B-port Write, B-Port Address Counter and the PN Generator (see page 31). With all these features enabled the memory is written with random data. Then repeat commands '6', '7', '8' and 'P' to disable all the features again (do not use 'R' as this will clear the memory).

Use the 'F' command to see how the accumulator will respond. Although both ports of the BRAM are enabled, the address to the B-port remains disabled and location zero of the memory containing one random value is applied to the multiplier. The address to A-port is incrementing and this applies various random values to the other input of the multiplier. This in turn means that the output of the multiplier will be a series of random values which will cause the accumulator to do all kinds of things!

Hint - You can try programming the memory several times or could even leave PN generator enabled with new pseudo random values continuously being written into the memory. Just remember to observe only the difference in current when the accumulator is enabled and dismiss the increase in total current caused by the PN generator and memory writing..

```
>f
BRAM  Accumulator
2FEF0 0102212100
359B7 01A91ACE90
2D7D9 02D27C7800
26DAF 046678EE10
1B5B9 02AEEED980
37C69 033311A9F0
175F2 01BB9258D0
3536B 0268DEB720
077CE 01F0916C40
1E32F 000B610A50
1BD4C FE4C3BE990
1E87A FC61BAE7F0
33952 FD293C00D0
342DA FDE72AF930
221C7 FFC76015C0
1848B FE41384210
02C61 FE14A81B00
11632 FCFD4E85E0
25264 FEACB2FBA0
0D186 FDDA4E5D40
```

Author's measurements at 50MHz only

Now use the commands '4', '5', '6' and 'M' to enable the multiplier, both BRAM ports and the A-port address counter in the same way that the 'F' command did but this time at the system clock rate. Note the supply current.

$$I_{CCINT} = 80.2\text{mA}$$

Use the 'A' command to enable and disable the accumulator and observe the increase in supply current.

$$I_{CCINT} = 85.3\text{mA} \quad (+5.1\text{mA})$$

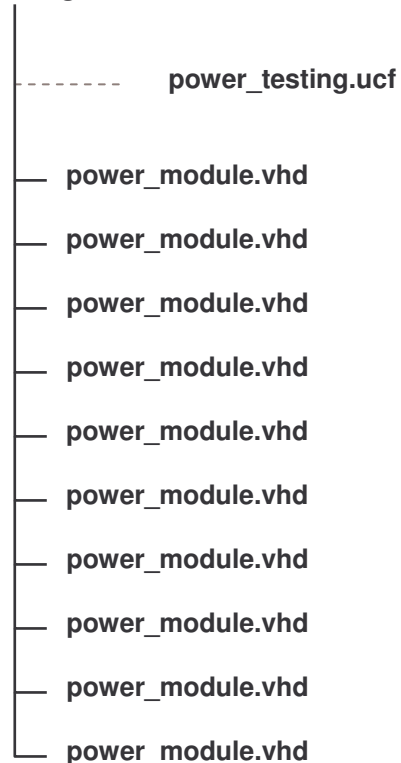
~0.5mA per Accumulator

In this experiment the outputs are clearly switching which must increase the supply current. However, this time the carry chain must be working less hard with the random data compared with the ± 1 pattern and overall we see a significant reduction in current. Random data is still a very extreme data pattern but it and the ± 1 are not so far removed from what may be seen in a DSP system. In other applications and accumulator may operate much more like a binary counter with far lower power consumption.

Design Files

The source design files of the reference design are shown below. The files contain comprehensive comments which you are advised to read in conjunction with this document. I do hope you will be expand your power experiments by modifying them yourself.

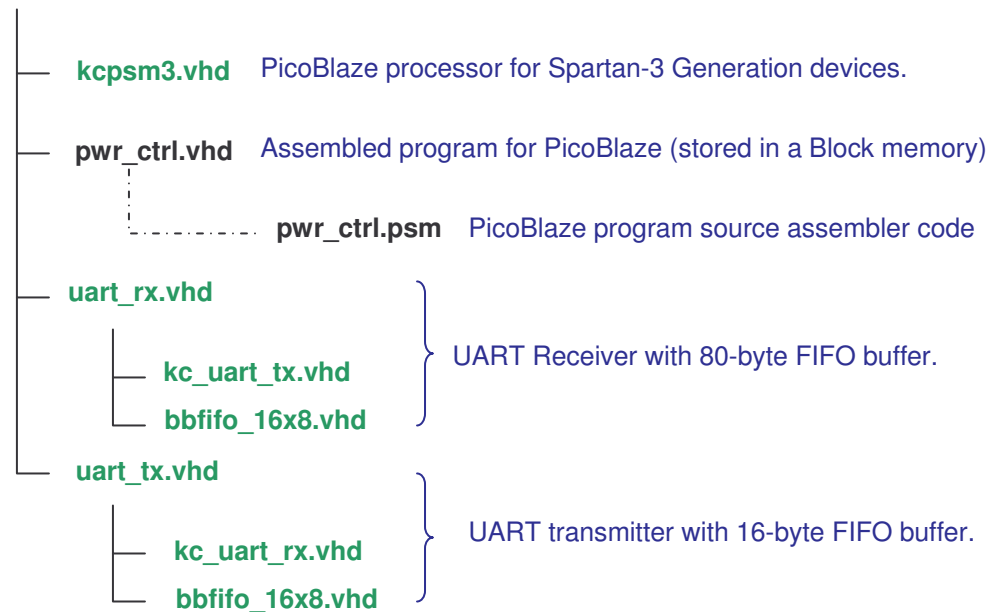
power_testing.vhd Top level file and main description of hardware.



power_testing.ucf I/O constraints file for Spartan-3A Starter Kit.
Timing specifications for maximum 50MHz system clock.
Constraints defining SUSPEND mode operation.

'power_module' is replicated 10 times to 'amplify' supply current measurements for greater accuracy.

power_module.vhd



kcpsm3.vhd PicoBlaze processor for Spartan-3 Generation devices.

pwr_ctrl.vhd Assembled program for PicoBlaze (stored in a Block memory)

pwr_ctrl.psm PicoBlaze program source assembler code

uart_rx.vhd

kc_uart_tx.vhd

bbfifo_16x8.vhd

} UART Receiver with 80-byte FIFO buffer.

uart_tx.vhd

kc_uart_rx.vhd

bbfifo_16x8.vhd

} UART transmitter with 16-byte FIFO buffer.

Note: Files shown in **green** are not included with the reference design as they are all provided with PicoBlaze download. Please visit the PicoBlaze Web site for your free copy of PicoBlaze, assembler and documentation.

www.xilinx.com/picoblaze

SUSPEND Mode

Suspend mode allows the Spartan-3A device to be placed into a state where the power dissipation is lower than even that experienced with the device completely static (no clock). In fact, it is even lower power dissipation than an unconfigured device (see page 13). In suspend mode the design configuration is retained along with all the states of the internal user logic so that the design can rapidly resume from the same state as it was when it went to sleep. Please see User Guide UG331 (chapter 18) and XAPP480 for full details but the following references design notes provide some general advice.

Preparing a Design for SUSPEND Mode – UCF File

Please take a look at the comments included in the User Constraints File (UCF) provided with this design. You will see that there is a specific constraint being used to declare that the suspend mode is to be enabled. The SUSPEND pin is a dedicated input on the device but will only be enabled if specified. The AWAKE pin is used to indicate if the device is active (AWAKE) or in suspend mode (asleep). However, the AWAKE pin is also a user I/O so the use of this constraint in the UCF file enables the ISE tools to reserve the AWAKE pin and check that you do not assign any user signals to it.

```
CONFIG ENABLE_SUSPEND = "FILTERED";

NET "led<0>" LOC = "R20" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 4 | SUSPEND = "DRIVE_LAST_VALUE";
NET "led<1>" LOC = "T19" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 4 | SUSPEND = "DRIVE_LAST_VALUE";
NET "led<2>" LOC = "U20" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 4 | SUSPEND = "DRIVE_LAST_VALUE";
NET "led<3>" LOC = "U19" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 4 | SUSPEND = "DRIVE_LAST_VALUE";

NET "init_b" LOC = "V13" | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 4 | SUSPEND = "DRIVE_LAST_VALUE";
```

You will also notice that associated with each output pin constraint is the definition of what each pin should do when the device is in suspend mode. In this design I have selected the option for the pins associated with the LEDs to retain their last known value.

Hint– Using this reference design you can easily confirm that the last known value is preserved during suspend mode. The user LEDs LD0, LD1, LD2 and LD3 have corresponding direct connections to the slide switches SW0, SW1, SW2 and SW3. When the design is active the AWAKE LED will be illuminated and the user LEDs should simply reflect the settings of the switches. When the SUSPEND switch (SW4) invokes suspend mode the AWAKE LED will be off but the user LEDs should be seen to retain their old state even when the slide switches are operated. When the SUSPEND switch is operated to return the device to active mode, AWAKE will illuminate and the user LEDs will adjust to any reflect the current switch positions. In a similar way the INIT LED normally blinks once per second and you should be able to invoke suspend mode such that the INIT LED remains either on or off during the suspend state.

Preparing a Design for SUSPEND Mode – Design

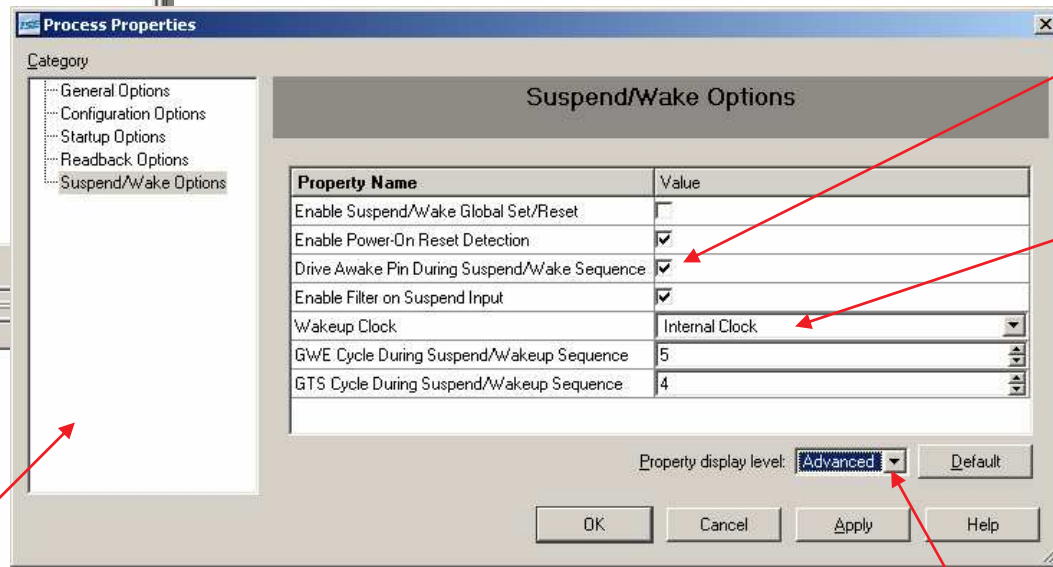
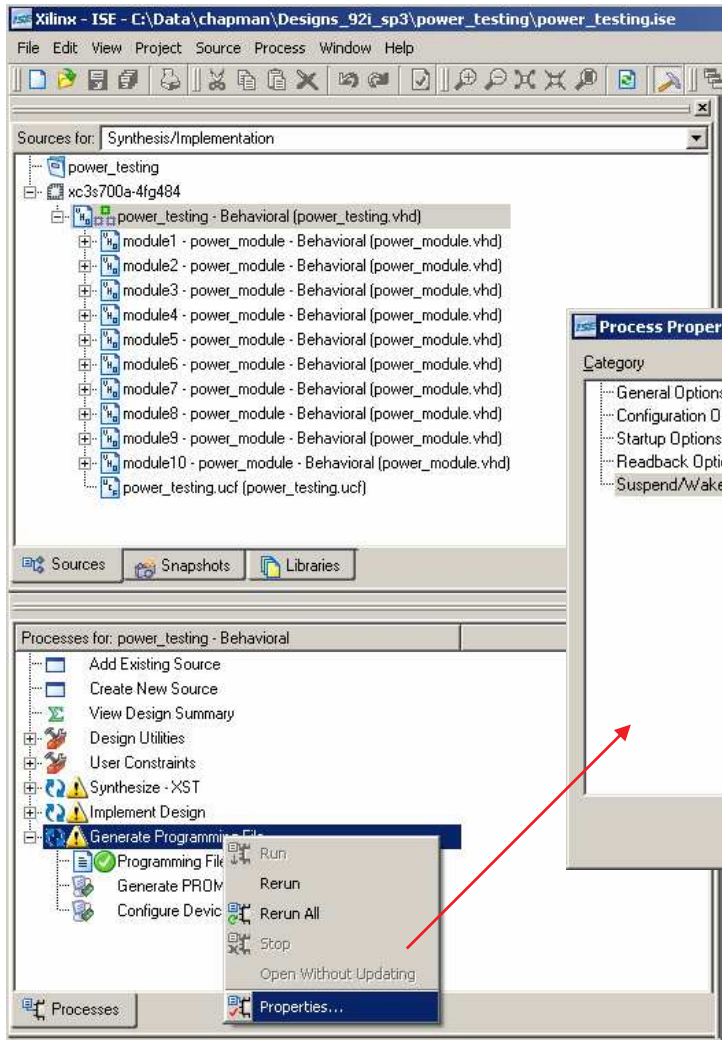
A designer should be aware that entry to, and exit from, suspend mode is essentially an asynchronous event and a designer should take reasonable precautions to ensure that the design will operate reliably. Particular consideration should be given to ensuring that the logic states which will be preserved during suspend mode all pertain to the same clock cycle if this is important (i.e. in a state machine). Therefore some form of pre-suspend and post-suspend handshaking with the user design may need to be considered. This simple reference design does not include such specific suspend control circuits.

SUSPEND Mode

Preparing a Design for SUSPEND Mode – Bit File Generation

Although the BIT and MCS files supplied with this reference design already have the suspend mode correctly enabled it is hoped that you will make your own modifications to the design to conduct further experiments. In this case you will need to ensure that your ISE project is correctly configured to invoke suspend mode in the way that will work with the Spartan-3A Starter Kit. Once again, please see User Guide UG331 (chapter 18) and XAPP480 for full details but these screen shots showing ISE v9.2 indicate the key settings used in the preparing of the provided files and these should also be your initial starting point.

Right click on 'Generate Programming File' and select 'Properties...'. Then chose 'Suspend/Wake Options'.



Drive Awake Pin During Suspend/Wake Sequence

Internal Clock

You may need to select 'Advanced' to see the options shown here.

Summary of Author's Measurements

This reference design is intended to encourage you to perform your own experiments and take your own supply current measurements. However, here is a quick summary of the observations made by the author. Please see previous pages for full details.

Quiescent

- $I_{CCINT} = 12.1\text{mA}$ Quiescent current of V_{CCINT} (1.2v) supply with device configured with this reference design
- $I_{CCAUX} = 11.5\text{mA}$ Quiescent current of V_{CCAUX} (3.3v) supply with device configured with this reference design
- I_{CCAUX} remains virtually constant even when the circuit is operating but can be reduced to 1.4mA using suspend mode.

Dynamic (additional current resulting from operating a function at the clock frequency indicated)

- $I_{CCINT} = +6.5\text{mA @ 6.25MHz}$ 10 Block Memories (BRAMs) plus ~1000 logic slices
- $I_{CCINT} = +45.3\text{mA @ 50MHz}$ (Based on 10 PicoBlaze Processors 'working hard' being a good representation for a general mixture of general logic and memory)
- $I_{CCINT} = +0.65\text{mA @ 6.25MHz}$ A single PicoBlaze 'working hard' (Which at 1.2v represents +0.8mW at 6.25MHz and +5.4mW at 50MHz).
- $I_{CCINT} = +4.53\text{mA @ 50MHz}$
- $I_{CCINT} = +0.1\text{mA @ 50MHz}$ A binary counter (power is virtually independent of number of bits)
- $I_{CCINT} = +1.0\text{mA @ 50MHz}$ When the port of a BRAM is enabled
- $I_{CCINT} = +0.150\text{mA @ 50MHz}$ The additional current caused by actually reading data from BRAM port that is enabled.
- $I_{CCINT} = +0.1\text{mA @ 50MHz}$ The additional current caused by writing data to a BRAM port that is enabled.

From which we learn that we should always consider disabling a BRAM port unless it is actively being written or read.

- $I_{CCINT} = +1.1\text{mA @ 50MHz}$ A dedicated 18×18 multiplier working hard.
- $I_{CCINT} = +0.5\text{mA}$ A 40-bit slice based accumulator working hard.
to 0.9mA @ 50MHz