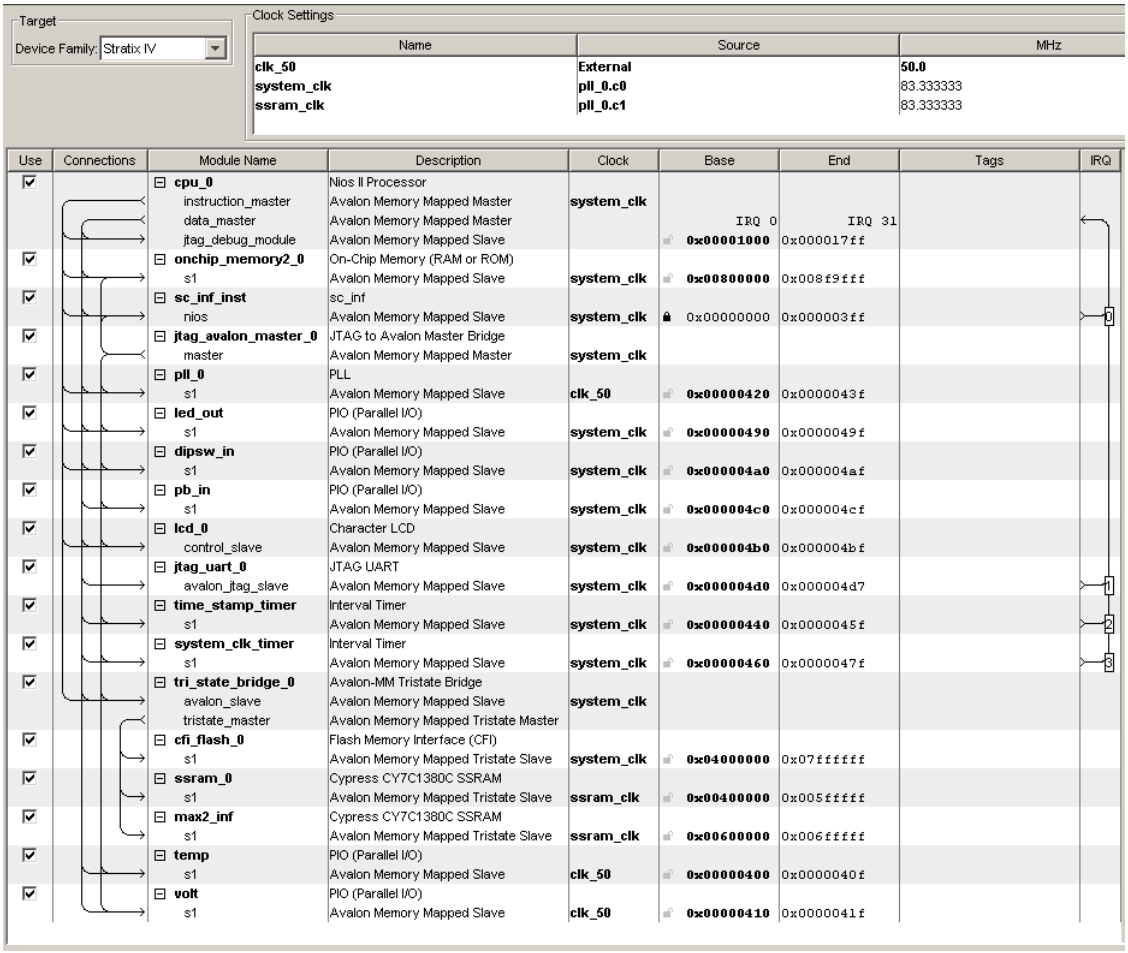
SSRAM_FLASH user guide



This project is just a connection to a lot of components.
There are 2 Avalon masters, CPU and JTAG-Avalon-Memory mapped Master.
Bellow them, there are

Component	instance name	Start address	End address
On chip memory		0x00800000	0x008f9fff
System Console Interface	(sc_inf_inst)	0x00000000	0x000003ff
PLL	(pll_0)	0x420	0x43f
LED interface	(led_out)	0x490	0x49f
Dipswitch interface	(dipsw_in)	0x4a0	0x4af
Push Button interface	(pb_in)	0x4c0	0x4cf
LCD interface	(lcd_0)	0x4b0	0x4bf
Temperature control	(temp)	0x400	0x40f
12V Voltage interface	(volt)	0x410	0x41f
CFI Flash interface	(cfi_flash_0)	0x04000000	0x07ffffff
SSRAM interface	(ssram_0)	0x00400000	0x005fffff

MAX-II register interface (max2_inf) 0x00600000 0x006fffff

I recommend to use System Console.
The system console can be called from SOPC Builder or command line.

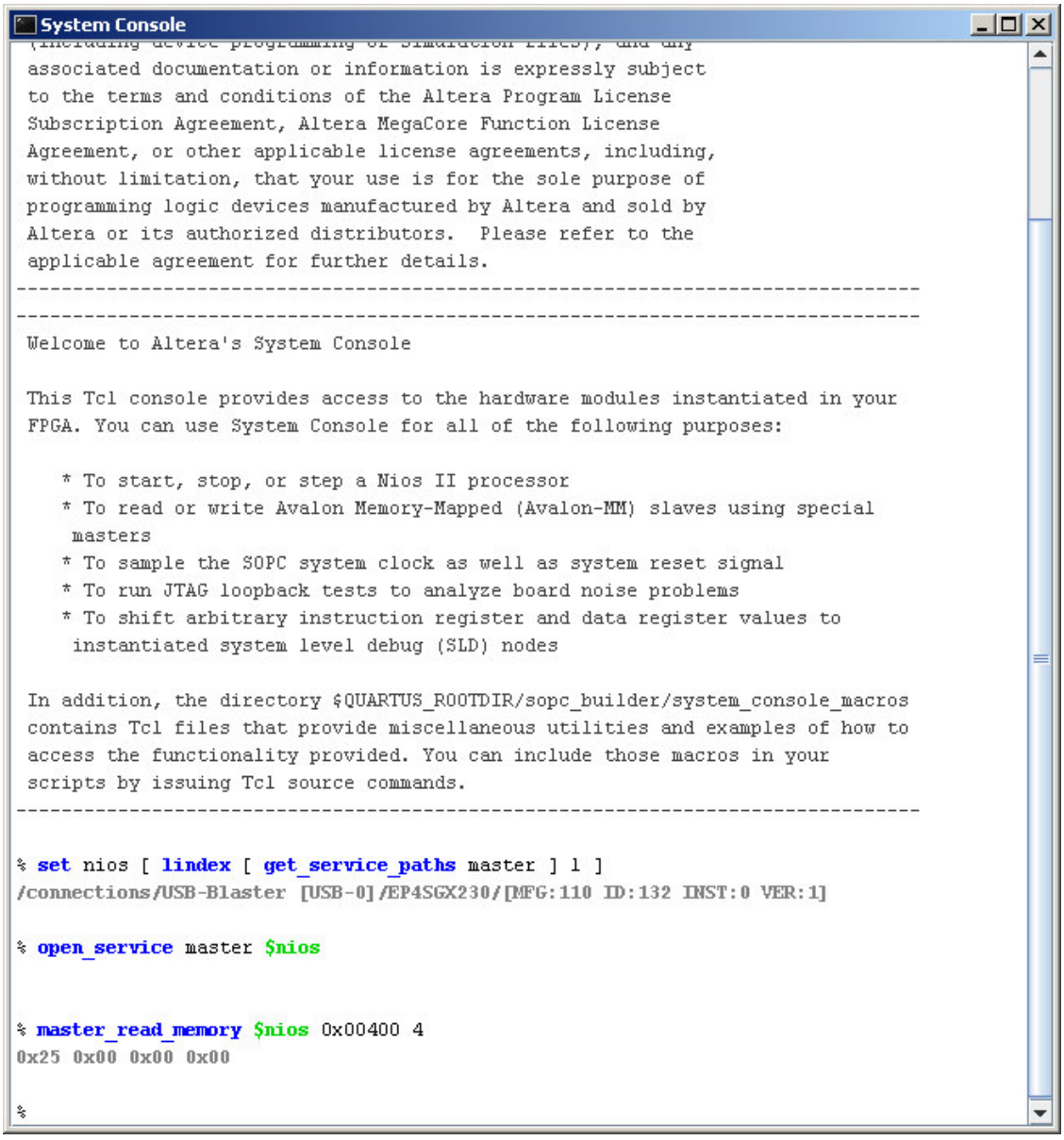
Here are the sample commands to use

1. set nios [lindex [get_service_paths master] 1]
2. open_service master $nios
3. master_read_memory $nios 0x00400 4
4. master_write_memory $nios 0x0040C [list 0x54]

3 and 4 is accessing the component.
Master_read_memory $nios [target_read_address] [byte_to_read]
So, the example is reading address 0x400 for 4 bytes.

master_write_memory $nios [target_write_address] [list write_data]
write_data is the hex data to write.
If you want to write 32 bits data, it will be [list 0x00 0x01 0x02 0x03]

For more information about the system console, please check online document for the system console.


System Console
(including device programming or simulation files), and any
associated documentation or information is expressly subject
to the terms and conditions of the Altera Program License
Subscription Agreement, Altera MegaCore Function License
Agreement, or other applicable license agreements, including,
without limitation, that your use is for the sole purpose of
programming logic devices manufactured by Altera and sold by
Altera or its authorized distributors. Please refer to the
applicable agreement for further details.
Welcome to Altera's System Console
This Tcl console provides access to the hardware modules instantiated in your
FPGA. You can use System Console for all of the following purposes:
* To start, stop, or step a Nios II processor
* To read or write Avalon Memory-Mapped (Avalon-MM) slaves using special
masters
* To sample the SOPC system clock as well as system reset signal
* To run JTAG loopback tests to analyze board noise problems
* To shift arbitrary instruction register and data register values to
instantiated system level debug (SLD) nodes
In addition, the directory $QUARTUS_ROOTDIR/sopc_builder/system_console_macros
contains Tcl files that provide miscellaneous utilities and examples of how to
access the functionality provided. You can include those macros in your
scripts by issuing Tcl source commands.
% set nios [lindex [get_service_paths master] 1]
/connections/USB-Blaster [USB-0]/EP4SGX230/[MFG:110 ID:132 INST:0 VER:1]
% open_service master $nios
% master_read_memory $nios 0x00400 4
0x25 0x00 0x00 0x00
%