Avalon MM Master Interface Simulator Compiler User Guide

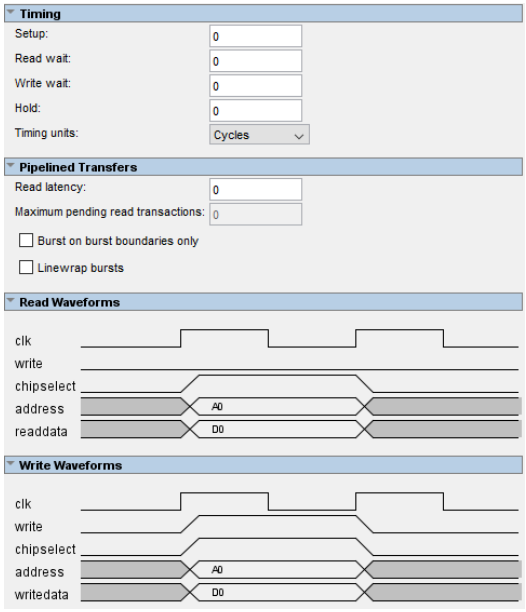
Zsolt Bagoly

December 3, 2019

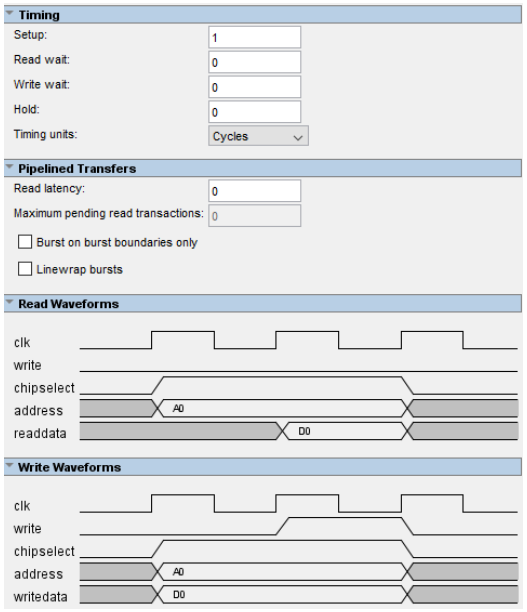
The HELP console output: *"avComp.exe help"*

```
I. Input arguments:
 - (1) *.av source to be compiled [by default: \"instruction.av\"]
 - (2) Verilog definition subfolder path [by default that is in the root]
 - Compiled file output: \"<source>.mem\" stored in the root directory.
 - Verilog definition file output: \"avsim_define.v\" stored in the root directory.
II. Acceptable Operating Codes (case-insensitive):
 - 1. nop: No operation for an Avalon cycle
 - 2. read: Reads the data from the specific address
 - 3. write: Writes the data to the specific address
 - 4. wait: Waiting until the specified cycles defined by the data
 - 5. load: Loading the timing parameters (see later)
III. Input Source Format:
 - 1. Instruction: <opcode> <hexadecimal address> <hexadecimal data>
 - 2. Comment: ; <any comments>
 - 3. Operating code, address and data must be separated by non-alphanumeric character e.g.
   white space.
 - 4. Comment section is not mandatory at the end, use the ';' key if needed.
 - 5. It is valid to use single line comment without instruction
IV. Limits:
 - 1. Each line (including the comment) is limited to 100 character.
 - 2. 4 Byte address and data in hexadecimal format.
 - 3. Maximum value of program counter: 999.
V. Timing settings: 1 Byte format with the usage of LOAD operating code.
 - 1. data: <Hold><ReadLatency><WriteWait><ReadWait> (MSB --> LSB)
 - 2. address: 0x000000<Setup> (MSB --> LSB)
 - 3. Example: \"load 11 2233aa01 ; setting avalon timing parameters\"
 => Setup: 0x11, ReadWait: 0x01, WriteWait: 0xaa, ReadLatency: 0x33, Hold: 0x22
VI. Input Source Format Error Handling:
 - 1. The specific line of the compiled output will be commented out in case of any source
   error.
 - 2. Compiler is able to distinguish the 3 different type of errors: opcode, address, data.
 - There will be placed an 'X' key where the input error is occurred.
VII. Source Example:
 ; Initialization
 load 0 00020001 ; Timing parameters: ReadWait = 1, ReadLatency = 2
 read 5 0        ; get module status

 ; Setting the module I/O data
 write 0 1235fe  ; setting the dividend
 write 1 a12     ; setting the divisor
 write 2 1       ; starting the module
 write 2 0
 wait 0 5        ; waiting for completion
 nop 0 0         ; wait one more cycle

 ; Obtaining the results
 read 3 0        ; get quotient
 read 4 a12      ; get remainder

 ; Integer Division Module Address Mapping
 ;   0x00: 32-bit dividend (cpu write)
 ;   0x01: 32-bit divisor (cpu write)
 ;   0x02: start operation (cpu write)
 ;   0x03: 32-bit quotient (cpu read)
 ;   0x04: 32-bit remainder (cpu read)
 ;   0x05: bit 0: 1-bit ready (cpu read)
 ;   0x06: bit 0: 1-bit done_trg (cpu read/write)
```

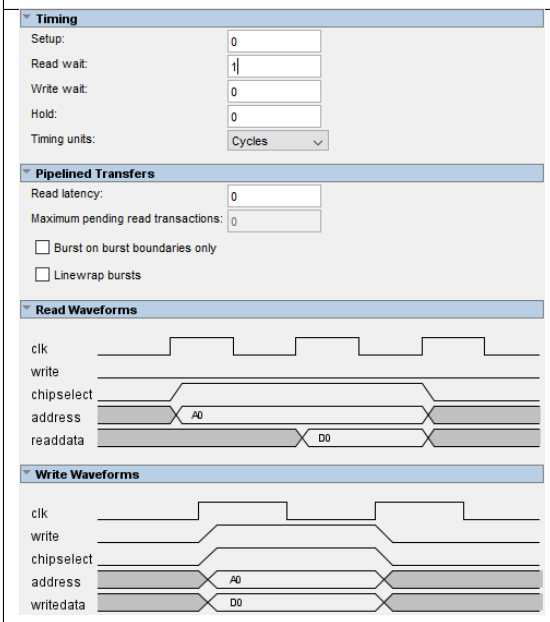
Avalon Memory-Mapped Slave Timing Parameters



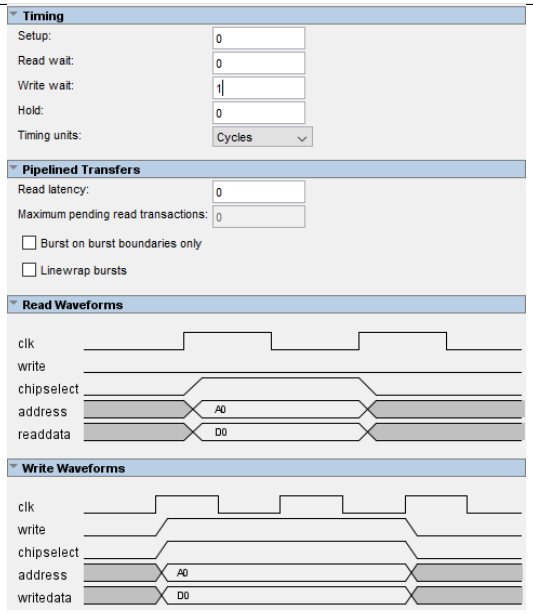
Default



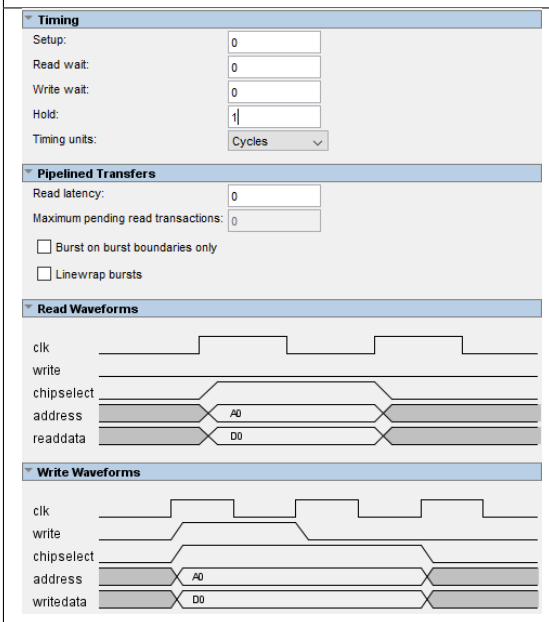
Setup = 1



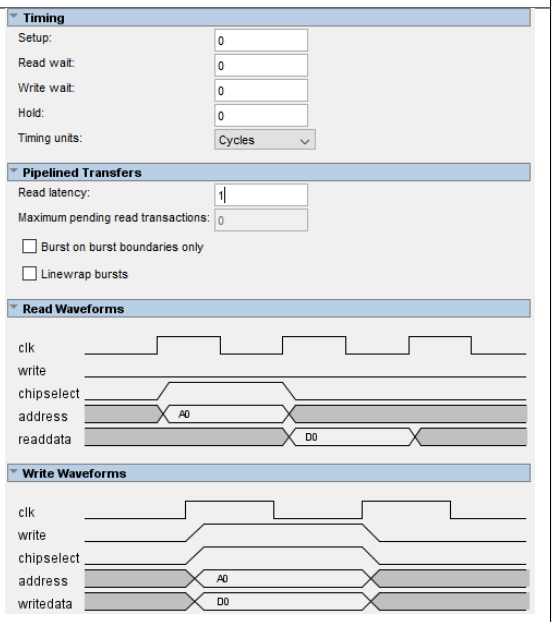
Read wait = 1



Write wait = 1



Hold = 1



Read Latency = 1