AHB Signals

The AHB specification defines a list of signals and defines how the different blocks in the system use those signals to communicate. Here, we'll quickly go through the different signals. Timing diagrams and more explanation are on the other AHB pages.

- **HCLK** The clock is an input to all elements in an AHB system and is assumed to come from some external clock generator. All AHB logic is rising edge triggered. The AHB specification does not define any particular clock frequency for AHB, but implementations above about 200MHz are pretty uncommon. If multiple AMBA clock domains are necessary in the system, then it is necessary to build multiple AHB buses with asynchronous AHB2AHB bridges to connect them together. Note that it is allowed (and common) for devices to use other clocks - it is only their AHB interface that must use HCLK.
- **HRESETn** This active low signal is an input to all elements in AHB. It is typically asserted low for a few (recommended to be at least 16) cycles. It is suggested that devices which require more than one or two cycles of reset have this fact documented.
- **HADDR[31:0]** A 32 bit bus, output from the master, which indicates the address to be used for a transfer. It is used as an input to the decoder and to all slaves. The decoder typically uses the most significant bytes of the address to decode which slave to select. The slave uses the remaining bits to decode which address within the slave (eg a particular register or memory location). The arbiter is allowed to use the address information as part of the arbitration scheme, but this is relatively uncommon. The address should be aligned according to the transfer size (HSIZE), even for IDLE transfers. (ie Word sized accesses must have HADDR[1:0]=0b00.)
- **HSIZE[2:0]** This master output indicates the size of the read or write transfer to be performed. The protocol supports up to 1024 bit transfers, but masters & slaves typically use 8, 16 and 32 bit sizes (and 64/128 in higher performance systems with larger databus widths).
- **HTRANS[1:0]** An output from the master (input to slave) which shows the type of transfer to be done. The allowable types are Non-Sequential, Sequential, Idle and Busy. Idle means that no transfer will be performed. The system is required to respond to idle transfers with HREADY=1 and HRESP=OKAY. Busy cycles occur as part of a burst and indicate that the master is unable to perform the next transfer in the burst on the current cycle. Non-sequential indicates that this transfer is unrelated to previous transfers (ie a single transfer, or the first of a burst) and Sequential indicates the second or subsequent transfer of a burst. For sequential transfers, the control signals may not change value compared to the previous transfer and there are rules about how the address relates to the previous address (depending upon the type of burst).
- **HWRITE** An output from the master which shows the direction of data transfer (a write - from master to slave; a read - from slave to master)

- **HBURST[2:0]** The master generates this bus to tell the system about the kind of burst it is performing - a single transfer, or a fixed, incrementing (incr) or wrapping burst. It can also give information to the slave about the length of the burst (1, 4, 8, 16 or undefined length). HBURST may potentially be an input to an arbiter, also.
- **HPROT[3:0]**This allows the master to tell the slave characteristics of the transfer including whether it is privileged/non-privileged, opcode or data and about the cacheable and write buffer properties of the access. Not all slaves need to make use of this bus.
- **HWDATA[n-1:0]** The master output which contains the data it wants to write to an address in a slave. The databus width is not fixed by the protocol but is typically 32 or 64 bits.
- **HRDATA[n-1:0]** The slave output (input to the master) which contains data that is read from a slave.
- **HSELx** A set of system-specific decoder outputs, generated combinatorially from the decoder's HADDR input (ie no register should be used). It produces a group of select signals, one for each slave.
- **HRESP[1:0]** An output from the slave (input to the master) which indicates whether the transfer was succesful (the slave gives the OKAY response). Other responses are ERROR (there was a problem with the transfer), RETRY (the master should attempt the access again) and SPLIT. This allows the slave to perform a slow transfer offline, without locking up the bus and requires the master to retry the access when it is next granted the bus when the split access completes.
- **HREADY** This is an output from each slave which shows when the transfer is completed. Holding HREADY low is effectively wait-stating the transfer. Note that slaves will also have an HREADY input so that they can see when previous transfers from other slaves have completed. The protocol does not enforce any quality-of-service or deadlock requirements, so if a slave holds HREADY permanently low, it can deadlock the AHB system.
- **HBUSREQx** An output from a master, input to the arbiter, requesting that the master is granted control of the bus.
- **HGRANTx** An output from the arbiter, input to the master, indicating that the master has been given the bus.
- **HLOCK/HMASTLOCK** In a full AHB system, a master will output HLOCK to show that the following transfer is locked and the arbiter will produce a corresponding HMASTLOCK signal, with same timing as HMASTER. In an AHB-lite system, the master will often output HMASTLOCK directly (or will need a small amount of additional logic to do so). A series of locked transfers are atomic, which means that no other transfer from an other master may occur during the locked sequence - the arbiter should not change grant while this happens. This is typically used to implement mutexes or semaphores for control of shared resources in multi-master systems.
- **HMASTER[3:0]** An arbiter output, in a full AHB system, showing the number of the currently selected master. This is used to control the muxes which route the selected master to the slaves and is also used by SPLIT-capable slaves to record the number of the master which performed an access. The width of 4 bits allows 16 masters to be supported. Systems typically have far fewer than 16 masters.
- **HSPLIT[15:0]** An output from split-capable slaves, to show the arbiter which masters are currently waiting a response from that slave and so should not be granted until that split transfer is ready to be restarted.