PerfectVIPs

I2C PACKET GENERATOR

DATASHEET

OVERVIEW

Packet Generator is a software tool that will be running on a Host computer and generate traffic patterns.

I2C packet generator is a product that as a Master generates series of data transfers which can be used by emulator platform to generate traffic on DUT Slave’s interfaces. Also it can be used on Simulation platform to generate traffic on simulation environment on DUTs interface.

FEATURES

- It works on two types of configuration modes: Data Read and Data Write.
- The physical I2C bus which transfers the generated data follows below points.
    - The two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. The SCL & SDA lines are connected to all devices on the I2C bus.
    - SCL is the clock line. It is used to synchronize all data transfers over the I2C bus. SDA is the data line.
    - Each device is recognized by a unique address (whether it is a microcontroller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the device.
    - In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers. A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.
    - The I2C-bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. As masters are usually microcontrollers, let us consider the case of a data transfer between two microcontrollers connected to the I2C-bus.

BENEFITS

- These methods can be used for verification of most complex hardware design to simple hardware design
- Easy to use solution, plug and play type solutions
- Software packet generators are very cost-effective solutions, they are cheaper compared to high license costing software products available in the market.
- Creates beautiful Emulation environment, which can mimic many simulation verification scenarios.
- Detect bug in Pre-silicon phase, which can save millions of dollars of re-spinning silicon cost.
- Post-silicon also software portion can be used for validation
- Help to build a parallel structure to simulation to find more design bugs quickly.
- The overall runtime can be reduced to as much as 10 times than long SOC simulations. This can speed up TAPE OUT of the chip.
- Scoreboarding and traffic analysis can be done very well in the Software solution.

Below is an example of I2C bus configuration using two microcontrollers.

- This example highlights the master-slave and receiver-transmitter relationships found on the I2C-bus. Note that these relationships are not permanent, but only depend on the direction of data transfer at that time.
- The transfer of data would proceed as follows:
    - Suppose microcontroller A wants to send information to microcontroller B.
- microcontroller A (master), addresses microcontroller B (slave)
- microcontroller A (master-transmitter), sends data to microcontroller B (slave-receiver)
- microcontroller A terminates the transfer.
    - If microcontroller A wants to receive information from microcontroller B.
        - microcontroller A (master) addresses microcontroller B (slave)
        - microcontroller A (master-receiver) receives data from microcontroller B (slave-transmitter)
        - microcontroller A terminates the transfer
- Even in this case, the master (microcontroller A) generates the timing and terminates the transfer.

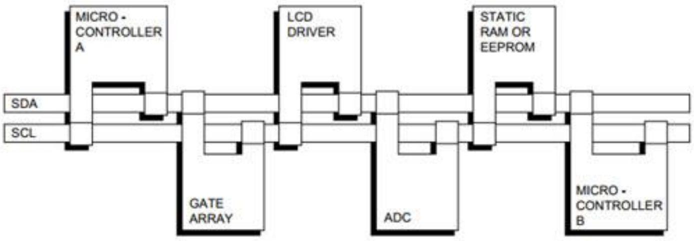

Figure 1: Example of an I2C bus configuration using two microcontrollers

Tools & Technologies: Verilog, SystemVerilog, C, DPI, UVM
EDA Tools
Emulators

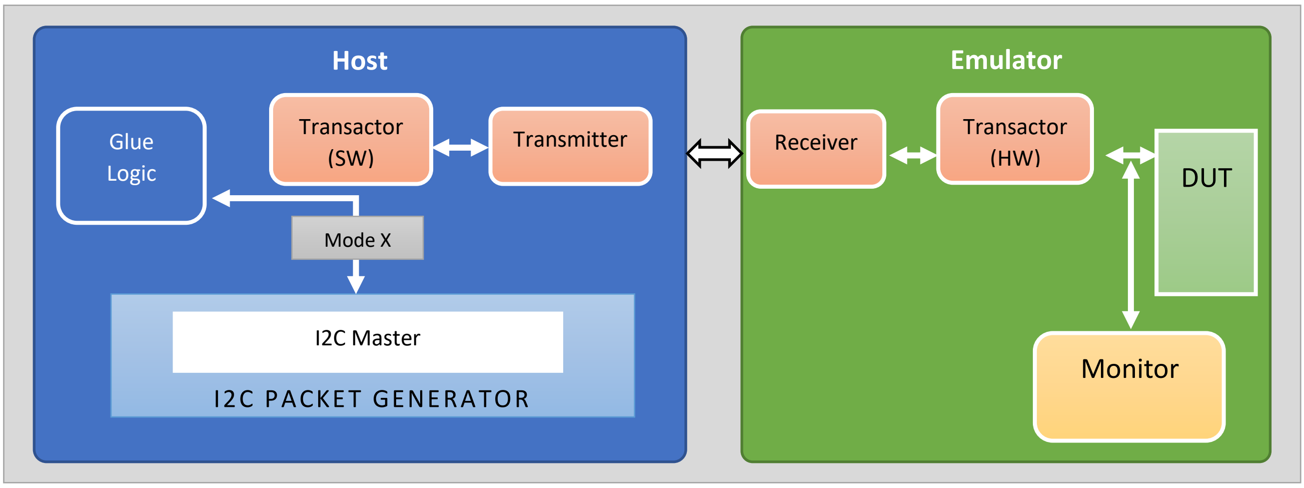

Figure 2: PCIe PACKET GENERATOR in EMULATION PLATFORM

CONFIGURATIONS

There are two configuration modes for data transfer:

- Data Read – Mode 1
- Data Write – Mode 2

DATA TRANSFER

Data transfers follow the format shown in Figure 3. After the START condition (S), a slave address is sent. This address is seven bits long followed by an eighth bit which is a data direction bit (R/W) — a 'zero' indicates a transmission (WRITE), a 'one' indicates a request for data (READ) (refer to Figure 4). A data transfer is always terminated by a STOP condition (P) generated by the master. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition (Sr) and address another slave without first generating a STOP condition. Various combinations of read/write formats are then possible within such a transfer.

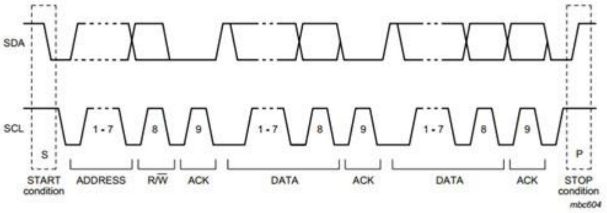

Figure 3: A Complete I2C Data Transfer

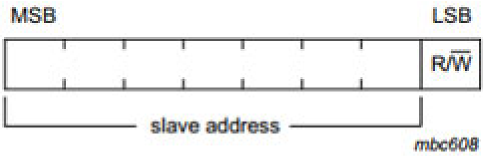

Figure 4: The 1st byte after the START procedure

Data transfer formats are:

- Master-transmitter transmits to slave-receiver. The transfer direction is not changed (see Figure 5). The slave receiver acknowledges each byte.

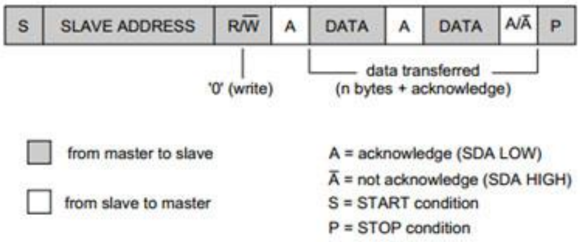

Figure 5: A master-transmitter addressing a slave receiver with a 7-bit address (the transfer direction is not changed)

- Master reads slave immediately after first byte (see Figure 6). At the moment of the first acknowledge, the master-transmitter becomes a master-receiver and the slave-receiver becomes a slave-transmitter. This first acknowledge is still generated by the slave. The master generates subsequent acknowledges. The STOP condition is generated by the master, which sends a not-acknowledge (A) just before the STOP condition.

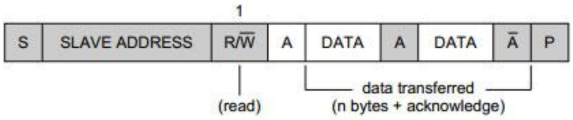

Figure 6: A master reads a slave immediately after the first byte

Combined format (see Figure 7). During a change of direction within a transfer, the START condition and the slave address are both repeated, but with the R/W bit reversed. If a master-receiver sends a repeated START condition, it sends a not-acknowledge (A) just before the repeated START condition.

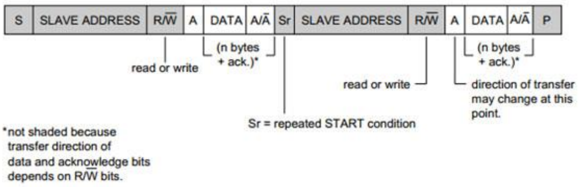
S
SLAVE ADDRESS
R/W
A
DATA
A/A
Sr
SLAVE ADDRESS
R/W
A
DATA
A/A
P
read or write
(n bytes + ack.)*
read or write
(n bytes + ack.)*
direction of transfer may change at this point.
Sr = repeated START condition
*not shaded because transfer direction of data and acknowledge bits depends on R/W bits.

Figure 7: Combined Format