Abhishek Vashist

Design of Computer System: EEEE-621

Lab-12

Objectives:

- Implement a 4-core system using the processor designed in the lab-11.
- Design a communication protocol between master and slave cores and evaluate the performance by performing a matrix multiplication.
- Emulate the working of processor on FPGA board

4-Core processor and communication protocol:

The processor is instantiated three times as Slave cores and the main processor is called the Master core. Master core has the access to the input and output peripherals of the FPGA board. Master distributes the given task to the other three slave cores and when they are done they send the result back to the Master core.

The abovementioned communication between the master and the slave cores is implemented using the handshake communication protocol, which is discussed in more detail in the next section.

Communication Protocol:

The communication happens when the master core wants to send data to slave cores and when the slave cores sends the manipulated data back to the master core. First we will discuss the master to slave communication protocol.

Master to Slave:

1) The Master first sends a request signal to the slave core before it can send any data and then waits for an acknowledgement form the slave core
2) Once an acknowledgement is received by the Master it sends the data along with a valid signal
3) The Slave core first checks for the valid signal from the Master to ensure that the data is valid and then it can store the data send by the Master.
4) After Slave core has received the data it sends an acknowledgement to the Master core indicating successful reception of the data.

The above protocol is repeated till the Master has data to send to the Slave cores.

In my design the data from the Master can be send to all the Slave cores at one time i.e. the Master to Slave communication is broadcast in nature.

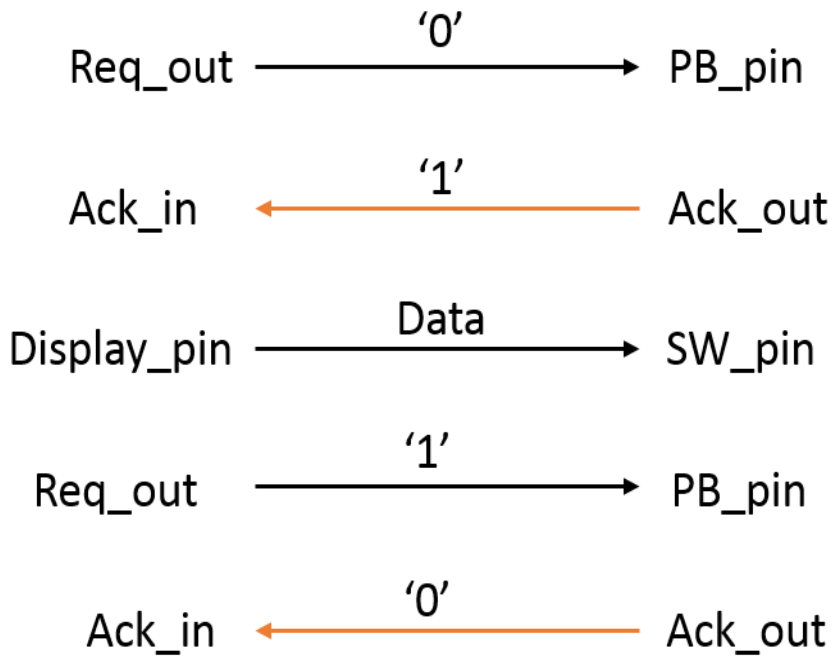


Figure 1. Master to Slave handshake protocol

Slave to Master:

1) The Master core first sends the slave ID from which it wants to receive the data from.
2) The Slave core that matches the ID sends a request to Master to indicate that it is ready to send the data
3) Master after receiving the request from the slave core sends an acknowledgement to the slave core that it is ready to receive the data
4) The Slave after receiving the acknowledgement from the Master sends the data along with the valid signal
5) The Master first checks if the data is valid and then it stores the data that was sent by the slave core
6) The Master send an acknowledgement to Slave to indicate that it has successfully received the data
7) The above protocol is repeated till the Master has received all the data from the slave core
8) After Master has received the data it can then set the slave ID for the next slave it wants the data from. The above protocol is repeated again.

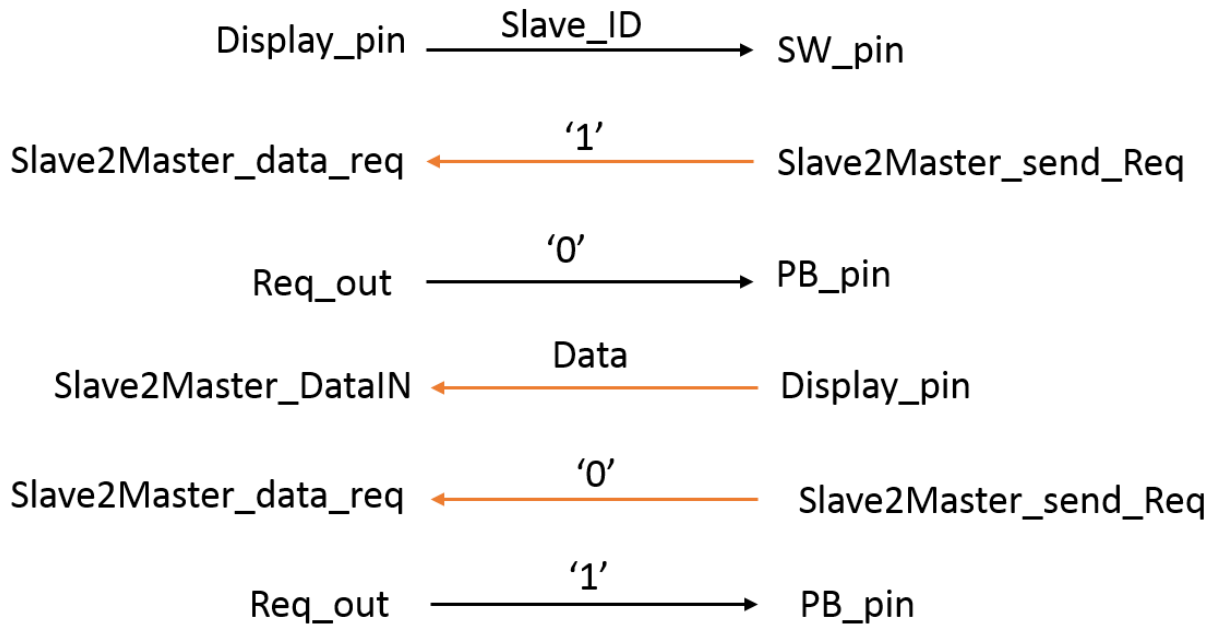


Figure 2. Slave to Master handshake protocol

Matrix Multiplication:

The communication protocol is implemented in assembly to perform multiplication of the two 4x4 matrices. Each row of the resultant matrix is computed by one of the core in the 4-core system. Here the Master computes the first row, Slave0 computes the second row, Slave1 computes the third row and Slave2 computes the fourth row of the resultant matrix.

Benchmark Results:

To evaluate the performance of the 4-core system we compute the total time taken (clock cycles) by the system to compute the output matrix once all the inputs are passed. For this additional performance counter logic is added in the processor that counts up the clock cycles once we initiate it. We further compare the performance of the designed 4-core system with the single-core system. The results are shown in figure 3.

Single core processor: 848 clock cycles

4-Core Processor: 1358 clock cycles

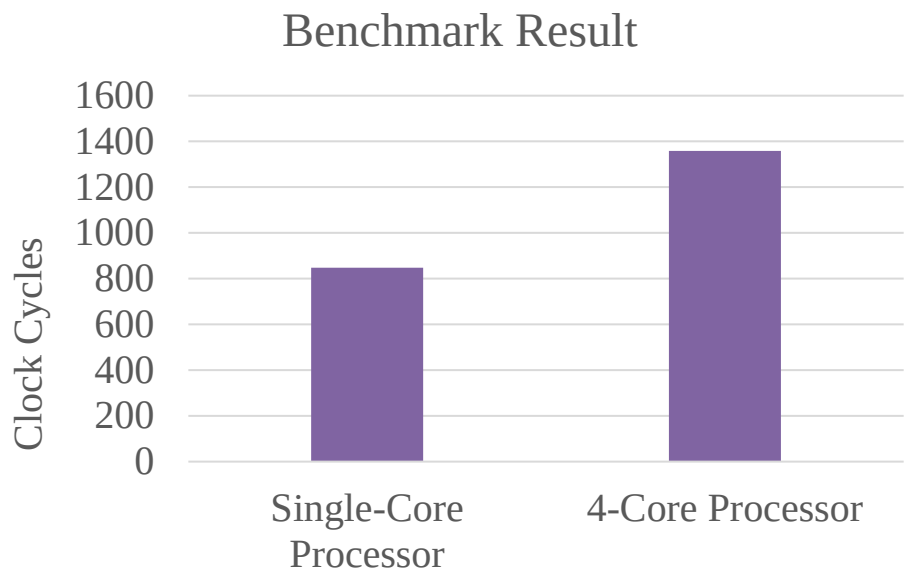


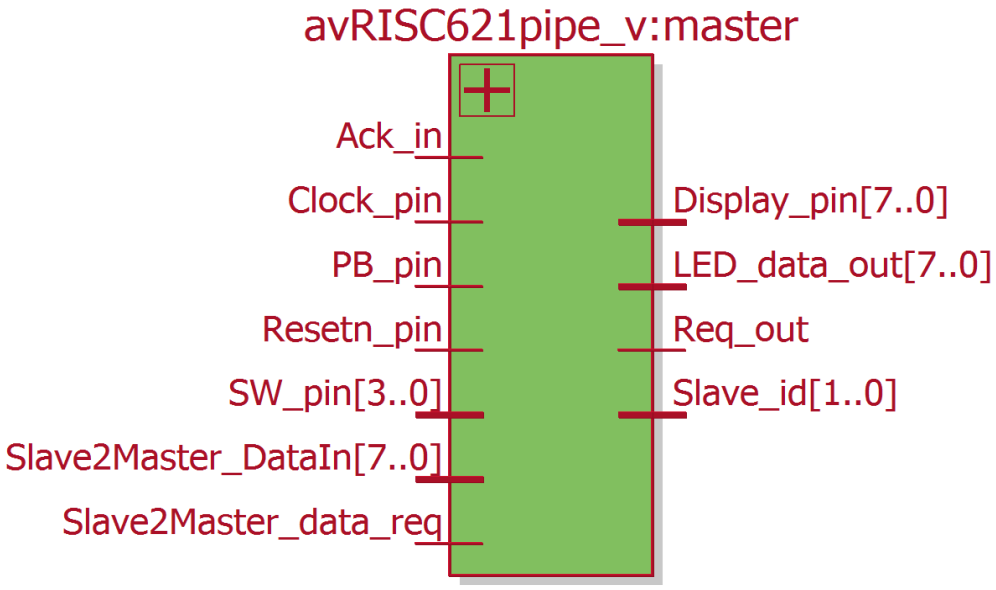
Figure 3. Benchmark result for 4-core and single-core system.

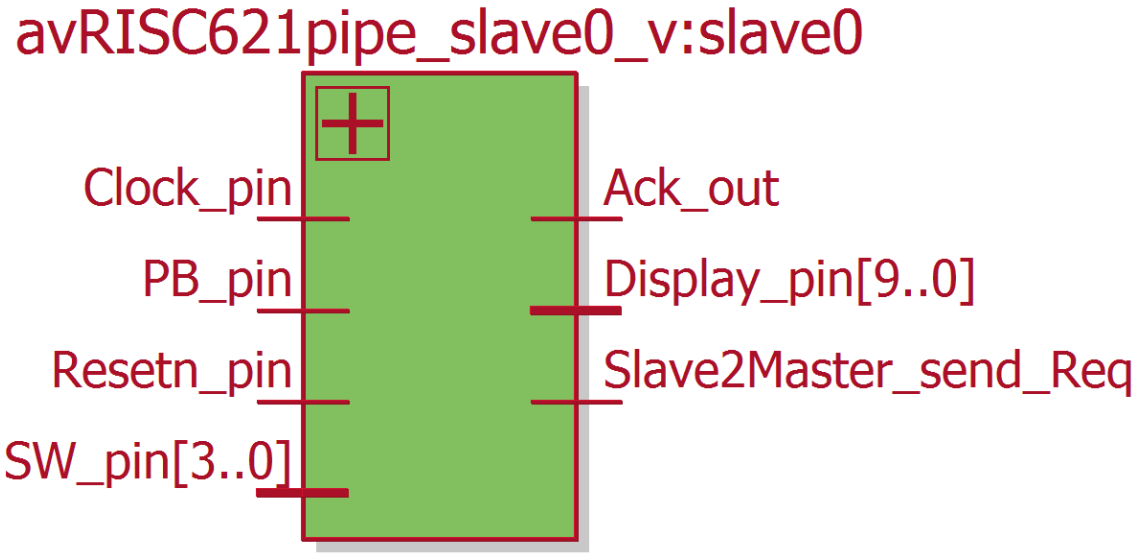
It can be seen that the 4-core system take more time to perform the matrix multiplication task when compared to the single core system. This is due to the additional communication overhead in the 4-core system.

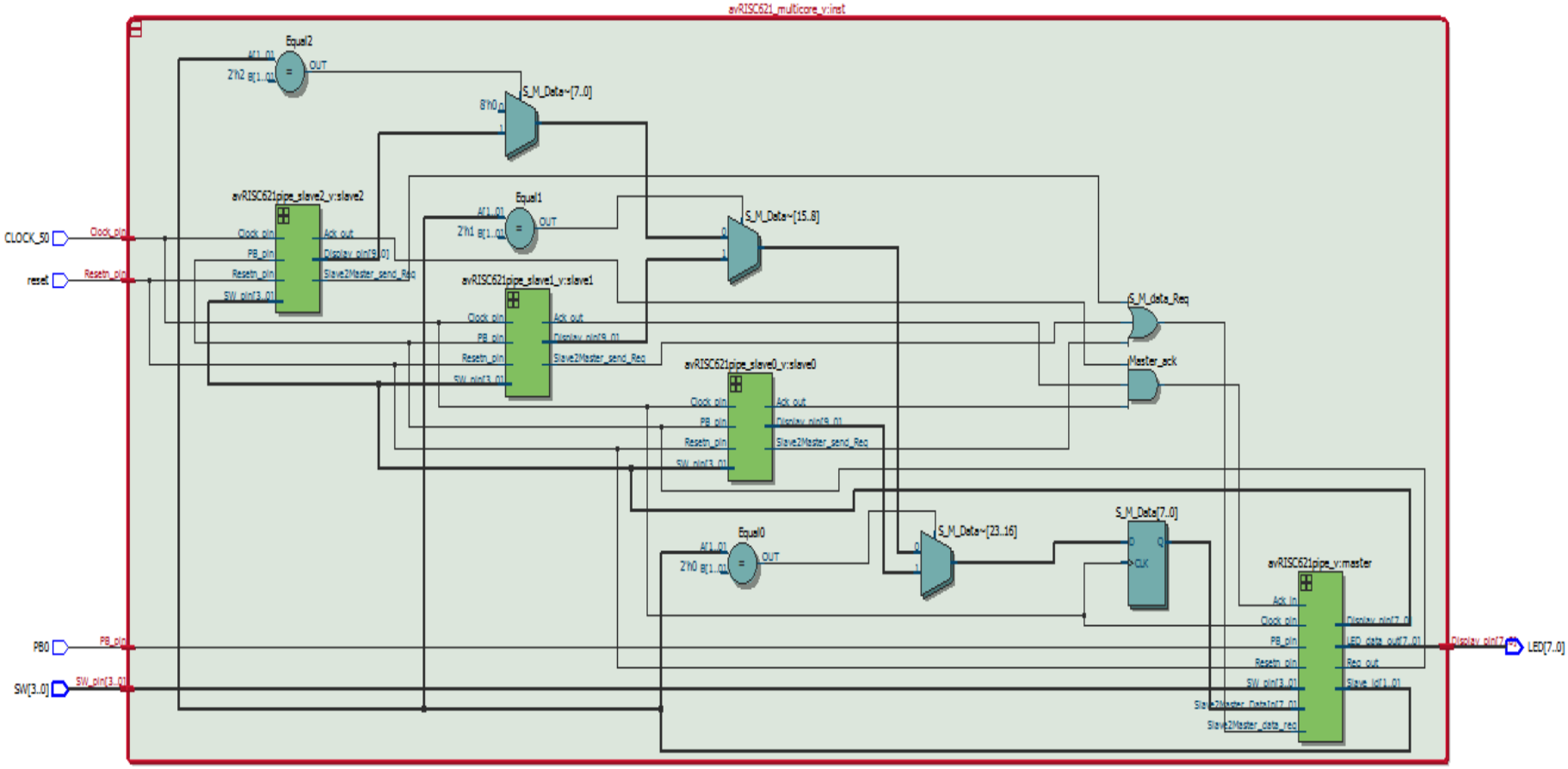
Communication overhead in the 4-core system: 510 clock cycles

APPENDIX:

Schematic views of the Master core, Slave core and the 4-core system are shown below.


avRISC621pipe_v:master
Ack_in
Clock_pin
PB_pin
Resetn_pin
SW_pin[3..0]
Slave2Master_DataIn[7..0]
Slave2Master_data_req
Display_pin[7..0]
LED_data_out[7..0]
Req_out
Slave_id[1..0]


avRISC621pipe_slave0_v:slave0
Clock_pin
PB_pin
Resetn_pin
SW_pin[3..0]
Ack_out
Display_pin[9..0]
Slave2Master_send_Req


avRISC621_multicore_v:inst
Equal2
Equal1
Equal0
S_M_Data~[7..0]
S_M_Data~[15..8]
S_M_Data~[23..16]
S_M_Data[7..0]
S_M_data_Req
Master_ack
avRISC621pipe_slave2_v:slave2
avRISC621pipe_slave1_v:slave1
avRISC621pipe_slave0_v:slave0
avRISC621pipe_v:master
CLOCK_50
reset
PB0
SW[3..0]
LED[7..0]