GH09.B.6.2dconv – 2 Dimensional Convolution

General Description

This design is a 2 dimensional convolution core in which an original matrix and a convolution matrix are passed in via an input, and the result matrix is passed out on an output port. The matrices are defined in terms of signed fixed point numbers of the form fx8.16 meaning there are 8 bits for the magnitude and 8 bits for the fractional part. The range, therefore, is $2^{(8-1)}$-1 (128) to -$2^{(8-1)}$ (-128).

Features

- 2d Convolution for unsigned fixed point fx8.8
  - 5 by 5 convolution matrix
  - 400 by 400 input (original) and output matrix

Block Diagram

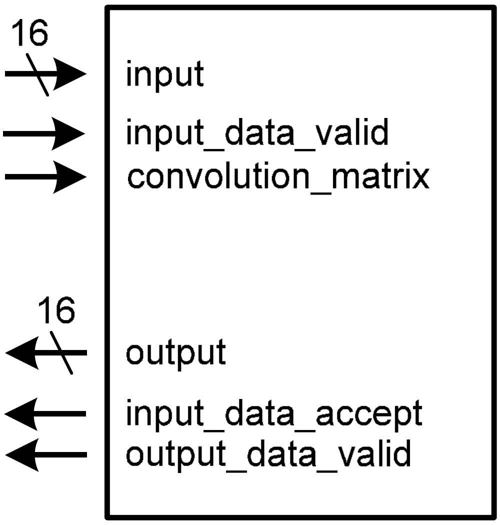

Figure 1 – Block diagram of the 2d convolution

Details

The 2d convolution is defined based on a mathematical equation, and the operation of the design is based on the input and output pin protocol.

Pins

The input and output protocol defines many of the pins further below. The remaining pins are defined as follows:

- input – this is a simple 16 bit port to pass in each fx8.16 matrix entry. The ordering is the first 25 inputs are for the convolution matrix, and the next 160000 are for the input matrix. [Optional] The ordering of inputs is by rows for the workload inputs.
- convolution_matrix – this signal indicates when the input contains values for the convolution matrix.
- output – this is the output 16 bit port that sends out the convolved bits of the matrix. [Optional] The ouputs are by rows in the same order as the inputs.

Chip Description

Equation

The execution of 2d convolution in discrete is based on two inputs: the convolution matrix (*h*) of size M and the input (original) matrix (*x*). The convolution operation is defined by the equation for all values of *r (rows)* and *c (column)* indices:

$$y[r,c] = \sum_{j=0}^{M-1}\sum_{i=0}^{M-1} h[j,i] \bullet x[r-j,c-i]$$

Basic operation

The equation, as specified above, and the input and output protocols define most of the design with the exception of the basic operation of the device. To execute a convolution the following steps are taken:

1. Reset the chip

2. Set the *convolution matrix* pin high to indicate that the next 25 input packets will constitute the convolution matrix

3. Send the convolution matrix through the input port as specified in the Input Protocol section (below)

4. Set the convolution matrix pin low to indicate that the input matrix data will now be sent

5. Send in the data on the input port pins

6. Once the convolution operation is complete, then repeat starting from step 2

Input Protocol

Inputs are received using a simple handshaking protocol. The pins of importance are the input (parallel port), the input_data_ready, the input_data_valid, the reset, and the clock. Figure 2 shows the basic communication setup between receiver (this design) and sender (external environment), and Figure 3 shows a waveform for a simple transfer between sender and receiver assuming the sender has one 8 bit packet to send (note that the input data width in the examples is 8 bits as opposed to 16 bits in the actual 2D convolution design shown in Figure 1 above).

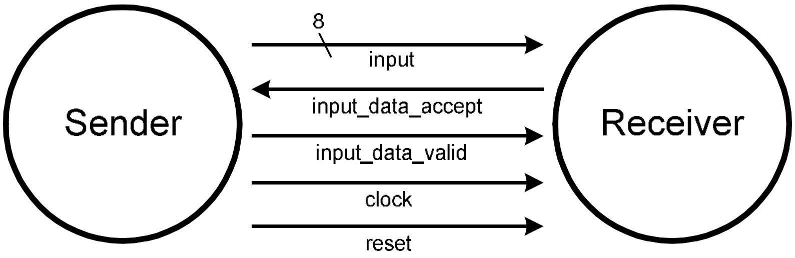

Figure 2 – The interface signals between the sender (the external environment) and the receiver (this design).

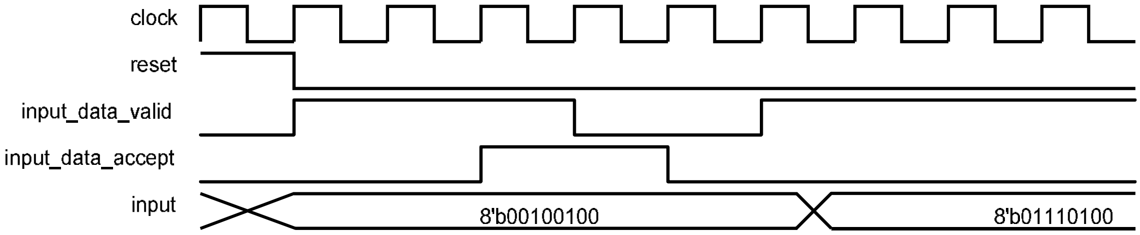

Figure 3 – a waveform for the communication between sender and receiver for one packet.

The assumption in this simple send receive protocol is that the shared clock between sender and receiver are synchronized. This simple interface removes much of the circuitry that would be needed to interface between devices.

Figure 4 and 5 show the finite state diagrams of both the sender and receiver.

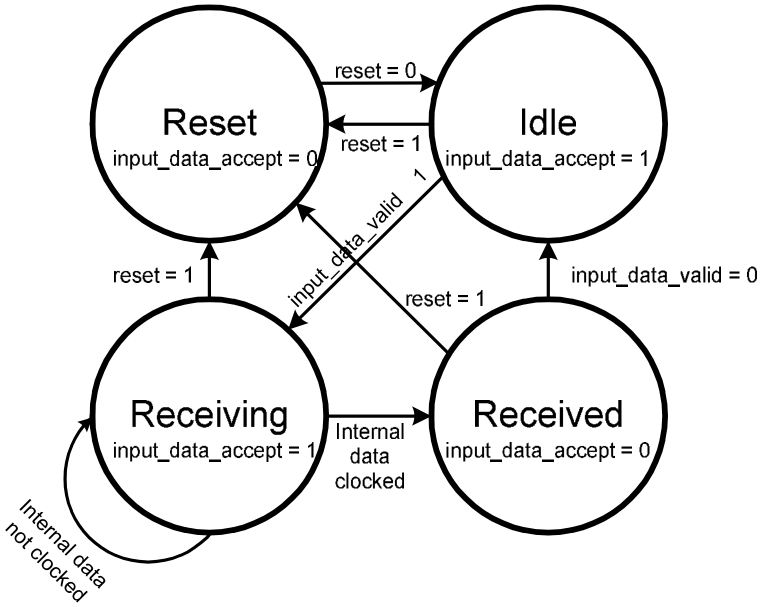

Figure 4 – Finite state machine for the receiver.

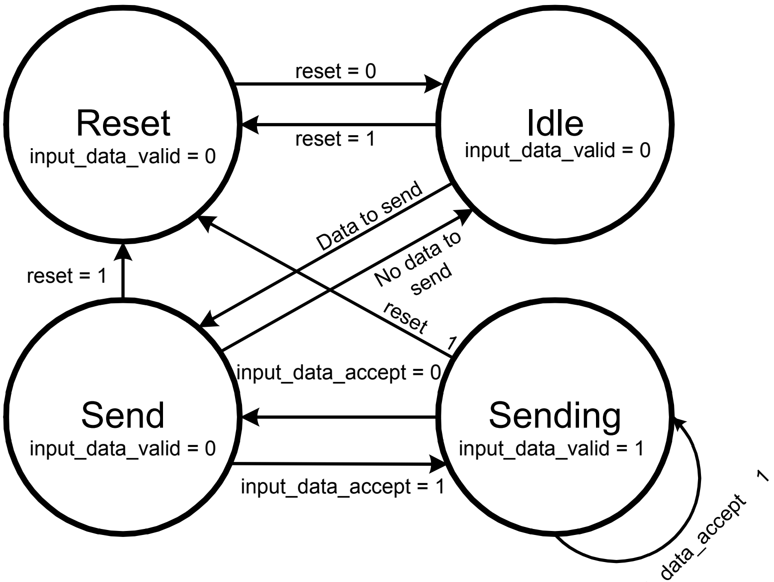

Figure 5 – Finite state machine for the sender.

Output Protocol

The output protocol is a very simple protocol. When the ouput_data_valid signal is high for one clock cycle, then the output bus contains a value. Figure 6 shows a waveform where three output values (val0, val1, and val2) are sent out of from the design.

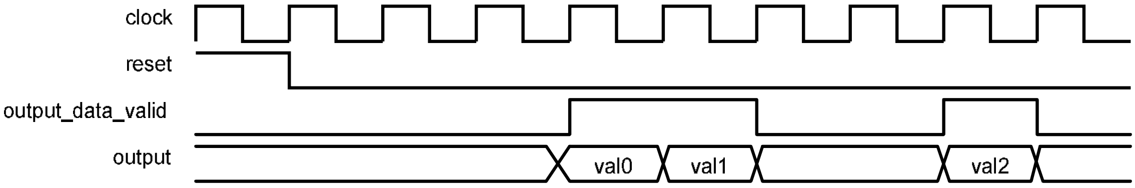

Figure 6 – A waveform showing the output being sent and the corresponding output_data_valid signal.