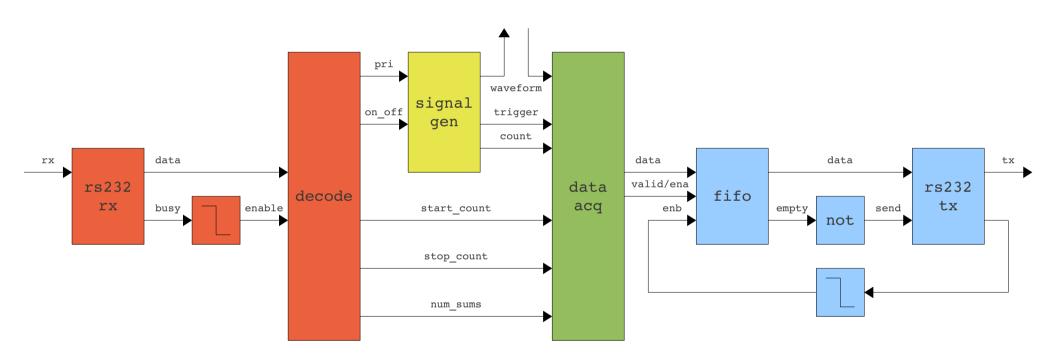
System Design



signal gen

- the signal generator block provides a number of functions.
 - Timing
 - PRF Trigger.
 - Master on/off of the entire system.
 - shared counter that data acq block uses.
 - Waveform
 - stores and generates the transmit waveform.
- there are two main components
 - Counter
 - RAM

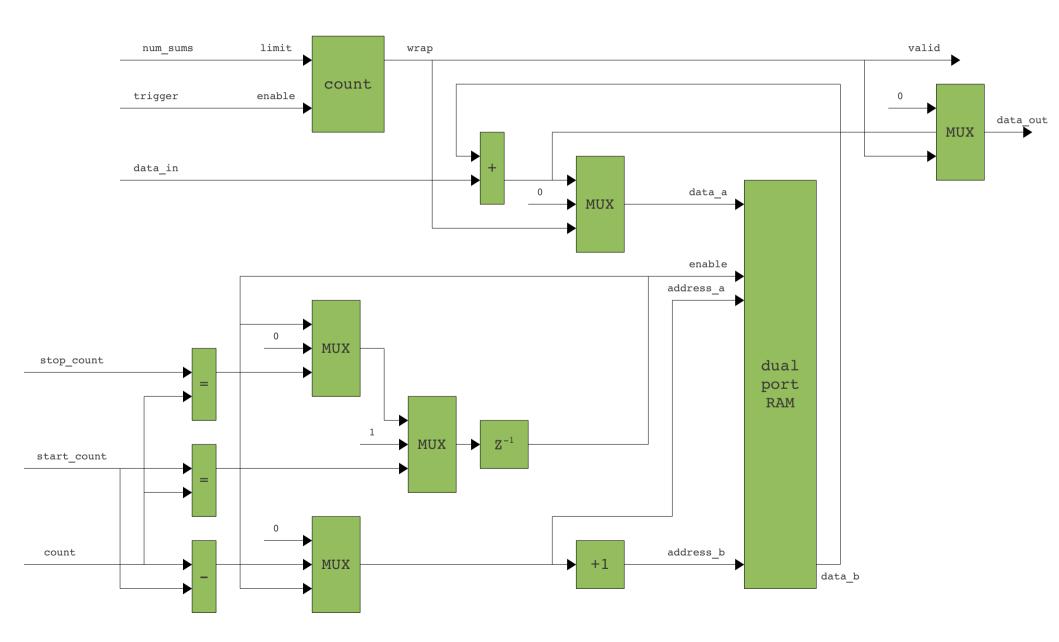
signal gen

```
pri
                               limit
                                                                      addr
                                             count
                                                                                    waveform
                                                                           wave
                                                                            RAM
                                                                      enable
      on_off
                               reset
                                                     RAM DEPTH
                 not
                                     count
                                                                                       count
module count (
                                                                                    trigger
                                             wrap
  input wire clock,
  input wire reset,
  input wire [20:0] limit,
  output wire [20:0] count,
                                                 module wave RAM (
  output wire wrap);
                                                    input wire clock,
                                                    input wire enable,
  reg [20:0] count int;
                                                    input wire [9:0] address,
  assign count = count int;
                                                    output reg [7:0] waveform);
  assign wrap = (count == limit);
                                                    reg [7:0] wave reg [1023:0];
  always@(posedge clock)
                                                    initial begin
    if (reset)
                                                      $readmemh("waveform.txt", wave reg);
      count int <= 0;</pre>
                                                    end
    else if (count == limit)
      count int <= 0;</pre>
                                                    always@(posedge clock)
    else
                                                      if (enable)
      count int <= count int + 1;</pre>
                                                        waveform = wave reg[address];
endmodule
                                                 endmodule
```

data acq

- For this application the primary purpose of the data acq block is to integrate data input.
 - start when the counter is equal to start count.
 - stop when the counter is equal to stop count.
 - dump the data when number of sums is reached.
- Primary components
 - counter to keep track of sums.
 - addition operation.
 - RAM to hold integrated data.
 - Muxes to route data.

data acq



```
assign wrap = (sum count == num sums);
always@(posedge clock)
  if (trigger)
                                                   data acq
    if (wrap)
      sum count <= 0;</pre>
    else
                                                                                            assign valid = wrap;
      sum count <= sum count + 1;</pre>
                                                                                           assign data out = (wrap) ? sum out : 0;
                             limit
                                                                                                                   valid
              num sums
                                                 wrap
                                      count
             trigger
                                                         assign sum out = data b + data in;
                             enable
                                                         assign data a = (wrap) ? sum out : 0;
                                                                                                                               data out
                                                                                                                        MUX
             data in
                                                                  0
                                                                                      data_a
                                                                        MUX
                                                                                                           dual port ram ram inst (
                                                                                                              .clock a(clock),
                                                                                      enable
                                                                                                              .data in a(data a),
                                               always@(posedge clock)
                                                                                   address_a
                                                                                                              .enable a(enable),
                                                 if (count == start count)
                                                                                                              .address a(address a),
                                                   enable <= 1;</pre>
                                                                                                           // .data out a(),
                                                 else if (count == stop count)
                                                                                                              .clock b(clock),
                                     MUX
                                                   enable <= 0;
 stop_count
                                                                                                              .enable b(1'b1),
                                                                                                dual
                                                                                                              .address b(address b),
                                                                                                port
                                                                                                              .data out b(data b));
                                                                                                RAM
                                                  MUX
start_count
                                                                                   address_b
 count
                                     MUX
                                                                         +1
                                                                                                        data b
                                  assign address a = (enable) ? count-start count : 0;
                                  assign address b = address a + 1;
```