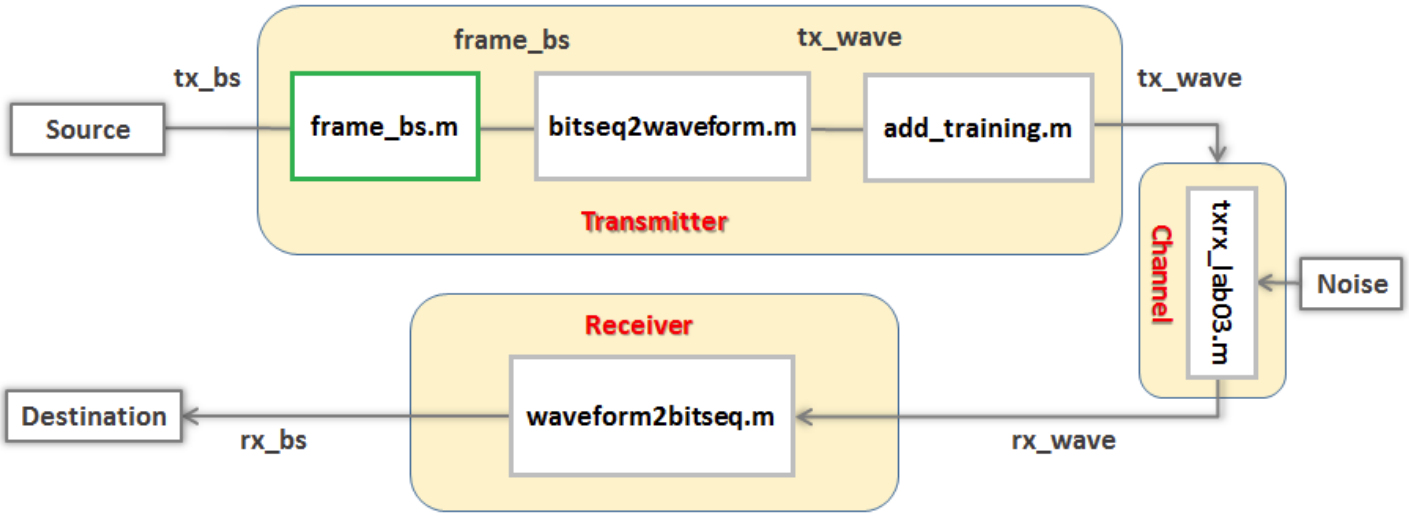
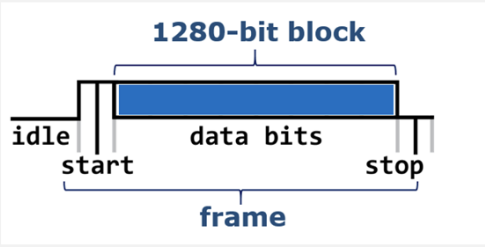
LAB 3 TASK 1 - FRAMING (1 point possible)

In this task, you will encapsulate a given bit sequence to a frame by working on the block, `frame_bs.m`, highlighted in green. Note that `frame_bs` is part of the function `format_bitseq` we used in the lab demo. Its function is to form the frame signal (without adding the training sequence).



The framing protocol is as follows:

1. The frame is 1282 bits long.
2. The frame contains one start bit with value "1".
3. The frame contains one stop bit with value "0".
4. The middle block of 1280 bits contains the data from the input bit sequence.
   - If the input bit sequence has a length less than 1280 bits, we use zero padding to fill the block.
   - If the input bit sequence has a length greater than 1280 bits, we discard the extra bits.



Help

The choice of the length of the data block is somewhat arbitrary. We have chosen 1280 here, as the length of an SMS message is limited to 160 characters, and we are using 8-bit ASCII codes to encode our characters (1280 = 160*8). If the input bit sequence has a length greater than 1280 bits, we could also break the bit sequence into multiple frames. This would probably be better, but we have chosen to discard the bits for the sake of simplicity.

Your job is to replace the lines between the comments
`% start of code for frame_bs.m`
and
`% end of code for frame_bs.m`
in the MATLAB window below with the code required to implement the function **`frame_bs.m`**.

You should not modify other lines of code. These lines are used to generate three random bit-sequences with different lengths and to check whether frames generated are correct. The lengths of the three bit sequences are random, but one is guaranteed to have a length less than 1280, one more than 1280 and one exactly equal to 1280.

```
bitseqs = bit_seq_gen();             % create three random bit sequences
frameseqs = cell(1,length(bitseqs)); % initialize holder for output

% this for loop runs the code for each of the three bit sequences
for c = 1:length(bitseqs),
    bs = bitseqs{c};

    % start of code for frame_bs.m
    frame = bs;
    % end of code for frame_bs.m

    frameseqs{c} = frame;
end

% generate plots
```

Unanswered

```
frame = zeros(1,1282); % takes care of zero padding and stop bit
frame(1) = 1;          % start bit
if length(bs) < 1280,  % copy bit sequence to frame
   frame(2:length(bs)+1) = bs;
else
   frame(2:1281) = bs(1:1280);
end
```

Run Code

Hide Answer *You have used 0 of 10 submissions*

Help

INSTRUCTIONS

This task has two steps.

Step 1: Run the code as presented

Click on the **Run Code** button to execute the MATLAB code in the window.

You will see a figure plotting the frames generated for the three bit sequences. Below the figure, you will see an error message generated because the line of code **`frame = bs`** is not a correct implementation of the framing protocol.

Step 2: Write the code implementing **frame_bs.m**

Replace the line

frame = bs

with your implementation of the framing protocol described above. Click on the **Run Code** button to check whether your implementation fully satisfies the protocol for all three randomly generated bit sequences. If everything is OK, the check_framing function will return the message

"OK! frame_bs appears to be correct."

Step 3: Submit your work

After you have finished your code, click on the **Check** button to submit your answer.